Controladores Lógicos Programáveis

Luiz Edival de Souza – edival@iee.efei.br

Fone: (0xx35) 3629 1181

Av. BPS, 1303 Itajubá –MG

FUPAI – 2001

Rua Xavier Lisboa, 27 Centro Itajubá-MG-CEP 37501-042

Fone: (0xx35) 3622-3477 Fax: (0xx35) 3622-1477

fupai@fupai.com.br - www.fupai.com.br

v4

SUMÁRIO

1 REVISÃO DE CONCEITOS LÓGICOS

Uma revisão da formulação apresentada pela Álgebra de Boole é importante para os usuários de circuitos à relés e controladores programáveis. O objetivo deste capítulo é revisar os conceitos básicos da lógica booleana visando a sua utilização em projetos de circuitos baseados em relés ou de programação do controlador programável.

POSTULADOS DA ÁLGEBRA DE BOOLE

1. $X = 0$ e $X = 1$ $\Rightarrow$ Qualquer variável e qualquer função, pode assumir somente dois valores representados por 0 e 1. Estes dois valores podem corresponder a duas situações ou grandezas físicas que se excluem mutuamente mas, necessariamente uma delas deve estar presente em qualquer instante.

2. $0 \bullet 1 = 1 \bullet 0 = 0$
3. $1 \bullet 1 = 1$
4. $0 \bullet 0 = 0$ $\Rightarrow$ Onde o ponto ($\bullet$) representa o operador lógico E ou "AND" do inglês. Pode-se em termos de contatos de relés associar o E a conexão em série de contatos;

5. $1 + 0 = 0 + 1 = 1$
6. $0 + 0 = 0$
7. $1 + 1 = 1$ $\Rightarrow$ Onde (+) representa o operador lógico OU ou "OR" do inglês. Pode-se em termos de contatos de relés associar o operador a conexão em paralelo de contatos;

8. $\overline{1} = 0$
9. $\overline{0} = 1$ $\Rightarrow$ Onde o sinal (–) sobre a variável significa negação.

VARIÁVEL E EXPRESSÃO BOOLEANA

Variável booleana é um literal que representa o estado de alguma coisa que possui somente dois estados: falso ou verdadeiro, aberto ou fechado, está presente ou não está presente, etc. Por exemplo, se um relé está energizado então podemos representar o estado do relé (energizado ou desenergizado) por uma variável X cujos valores podem ser somente 1 ou 0.

Expressão booleana é uma expressão que relaciona uma ou mais variáveis booleanas através dos operadores booleanos (E, OU e negação). Por exemplo, o motor deve ligar se a chave CH1 for

acionada e se a temperatura estiver acima de 40 °C. Neste caso atribuímos a uma variável M a representação do estado do Motor e escrevemos a seguinte expressão booleana:

M = CH1 . T onde:

CH1 = 0 para chave aberta e CH1 = 1 para chave fechada;
T = 1 se temperatura acima de 40 °C e T=0 se menor 40 °C
M = 1 motor ligado e M = 0 motor desligado.

1.1 TEOREMAS DA ÁLGEBRA DE BOOLE

A seguir são apresentados alguns dos teoremas usuais da Álgebra de Boole que quando convenientemente utilizados facilitam a simplificação de uma expressão complicada.

Num	Teorema
1	$0 \bullet X = 0$
2	$1 \bullet X = X$
3	$X \bullet X = X$
4	$X \bullet \overline{X} = 0$
5	$X \bullet Y = Y \bullet X$
6	$X \bullet Y \bullet Z = (X \bullet Y) \bullet Z = X \bullet (Y \bullet Z)$
7	$\overline{X \bullet Y \bullet Z} = \overline{X} + \overline{Y} + \overline{Z}$ Teorema de De Morgan
8	$\overline{f}(X, Y,..., Z,\bullet,+) = f(\overline{X}, \overline{Y},...., \overline{Z},+,\bullet)$
9	$XY + XZ = X(Y + Z)$ Obs: XY = X•Y
10	$XY + X\overline{Y} = X$
11	$X + XY = X$
12	$X + \overline{X}Y = X + Y$
13	$ZX + Z\overline{X}Y = ZX + ZY$
14	$XY + \overline{X}Z = XY + \overline{X}Z + YZ$
15	$XY + \overline{X}Z = (X + Z)(\overline{X} + Y)$

1.2 CIRCUITOS A CONTATOS

Examinaremos agora o relacionamento das expressões booleanas com circuitos a contatos. A partir das expressões booleanas podemos, através dos teoremas, simplificar os circuitos através da eliminação de redundâncias. Isto representa em termos de implementação menor custo, menos componentes, etc.

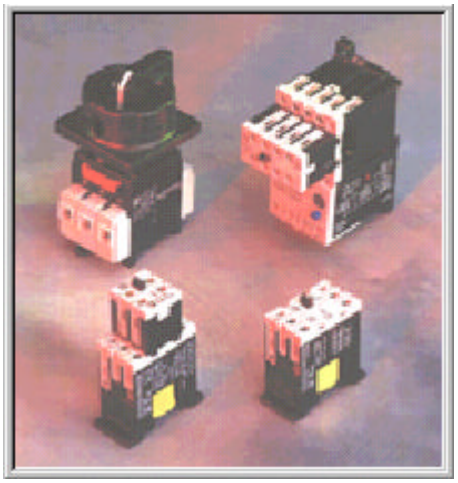
O contato aqui referenciado representa o estado de qualquer dispositivo do tipo liga/desliga utilizado em circuitos a relés. Um painel de relé, utilizado para controlar uma máquina ou um processo, pode ser visto como um conjunto de relés e um conjunto de dispositivos de entrada e saída, tais como, chaves, interruptores, válvulas, lâmpadas, contatores, etc. Por exemplo, para verificar se uma chave está ligada ou não, é preciso obter a informação de um contato do relé, ou para verificar se o motor está ligado é preciso, verificar se um contato auxiliar do contator do fechado (caso se use um contato NA - Normal Aberto).

Nos circuitos eletrônicos digitais, as entradas e saídas só podem estar em dois níveis de tensão, por exemplo, 0 V e 5 V. Nos circuitos a contatos, utilizamos dois estados - aberto e fechado, para representar o estado do contato. O estado da bobina do relé ou do circuito a contato é denominado energizado ou desenergizado. Assim sendo, podemos relacionar uma expressão booleana (valor 0 e 1) ao circuito a contatos (lógica por fios) e a variável booleana ao contato ou estado de chaves, botoeiras, etc. Portanto teremos:

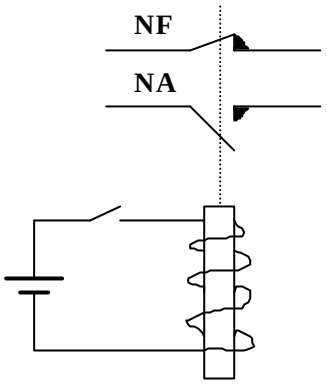
Expressão Booleana		***Circuito a contatos***
1	⇒	energizado
0	⇒	desenergizado

Variável Booleana		***Contato do relé***
1	⇒	fechado
0	⇒	aberto

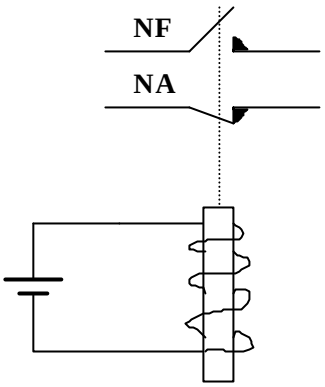
Relés



Contatores



Relé X - desenergizado



Relé X - energizado

De acordo com a nossa convenção podemos escrever a seguinte tabela:

Relé X	Contato NA	Contato NF
Desenergizado - 0	Aberto - 0	Fechado - 1
Energizado - 1	Fechado - 1	Aberto - 0

Onde observamos que : $NA = X$

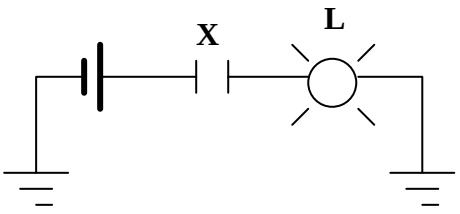
$NF = \overline{X}$

1.2.1 Exemplos de Circuitos a contatos

1) A saída de um circuito deve ser energizada se o relé X está operado e deve-se usar contato NA.

Solução:

A expressão booleana que expressa a solução deste exemplo é simplesmente : L = X, e o circuito a contatos pode ser desenhado como a seguinte figura.

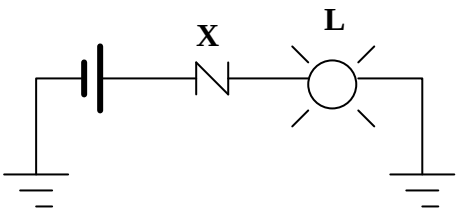


L = X

2) A saída de um circuito deve ser energizada se o relé X está inoperado e deve-se usar contato NF.

Solução:

O circuito abaixo atende esta exigência .

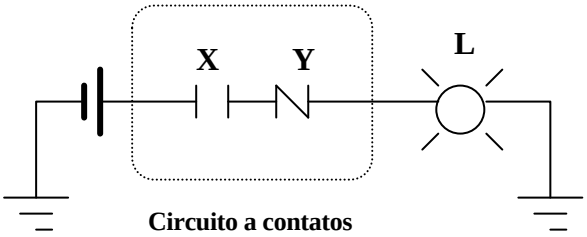


$L = \overline{X}$

3) A saída de um circuito deve ser energizada se o relé X está operado e o relé Y está inoperado.

Solução:

Observe que agora temos uma função E devido ao conectivo "e" na sentença de proposição do exemplo. A função E em circuitos a contatos pode ser obtida pela associação em série de contatos, como ilustrado abaixo.

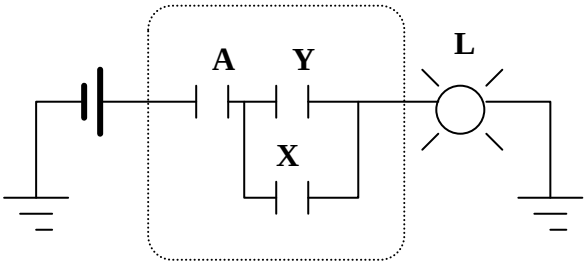


Circuito a contatos

$L = X \cdot \overline{Y}$

4) A saída de um circuito deve ser energizada se uma chave A for ligada e se o relé X ou o relé Y estiverem energizados.

Solução:



Circuito a contatos

$L = A \cdot (X + Y)$

1.2.2 Exercícios propostos

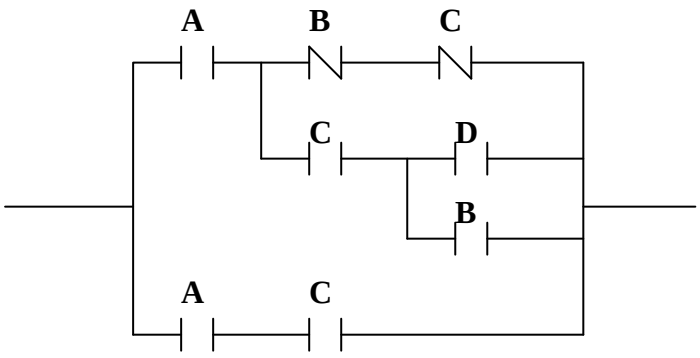
1) Desenhar os circuitos a contatos para realizar a lógica das seguintes expressões booleanas:

a) $L = A\,B + A\,C\,D + \overline{D}\,F + \overline{A}\,D\,\overline{F}$

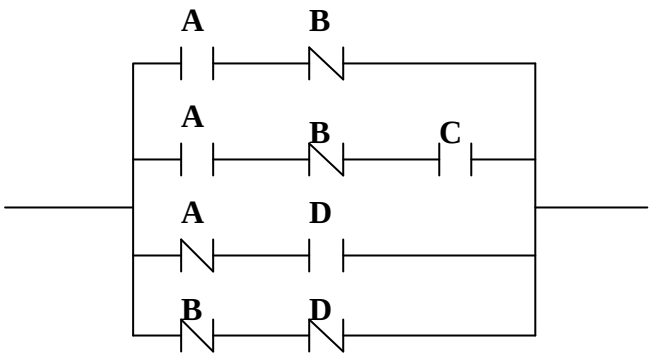
b) $L = (A + B)\ \overline{C} + (C + \overline{D} + F)(A + \overline{F})$

2) Simplificar os seguintes circuitos a contatos:

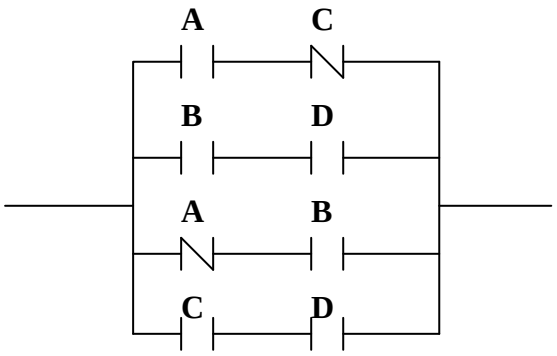
a)



b)



c)



3) Determinar a equação booleana, simplificada, para executar a seguinte lógica para ligar o motor de um ventilador :

1. Forno ligado e motor da esteira ligado, ou
2. Forno desligado e temperatura acima de 50 °C, ou
3. Forno desligado, relé X operado e motor da esteira ligado, ou
4. Motor da esteira ligado e temperatura acima de 50 °C, ou
5. Motor da esteira ligado, forno desligado e relé X inoperado, ou
6. Forno ligado, relé X inoperado e trem estacionado no local.

2 INTRODUÇÃO AO CONTROLADOR PROGRAMÁVEL

O critério de projeto para o primeiro controlador programável foi especificado em 1968 por uma divisão da GENERAL MOTORS CORPORATION. O objetivo inicial era eliminar o alto custo associado com os sistemas controlados a relés. As especificações iniciais requeriam um sistema de estado sólido com a flexibilidade do computador, capaz de suportar o ambiente industrial, ser facilmente programado e reprogramado, manutenção fácil e por último facilmente expansível e utilizável.

Devido ao intuito inicial de substituírem os painéis de relés no controle discreto, foram chamados de Controladores Lógicos Programáveis - CLP (*Programmable Logic Controllers - PLC*). Porém, atualmente, os controladores são bem mais complexos e não executam somente lógica do tipo E e OU, motivo pelo qual passaram a ser chamados apenas de Controladores Programáveis - CP.

Os primeiros controladores tinham pouca capacidade de processamento e suas aplicações se limitavam à máquinas e pequenos processos que necessitavam de operações repetitivas. A partir de 1970, com o advento da tecnologia de microprocessadores, os controladores passaram ter uma grande capacidade de processamento e alta flexibilidade de programação e expansão. Entre outras características citamos: a capacidade de operar com números, realizar operações aritméticas com ponto decimal flutuante, manusear dados e se comunicar com computadores. Desta forma, os CP's atuais podem atuar tanto em controle discreto, tais como, automação da manufatura, onde as máquinas apresentam ações automáticas e discretizada no tempo, como em controle contínuo, tais como, processos químicos e siderúrgicos, com características primordialmente analógicas.

O sistema utilizado para programar o controlador era um dispositivo dedicado e acondicionado em um maleta portátil, chamada de maleta de programação, de forma que podia ser levada para "campo" afim de alterar dados e realizar pequenas modificações no programa. O sistema de memória do controlador não permitia facilidades de programação por utilizar memórias do tipo EPROM.

Inovações no hardware e software entre 1975 e 1979 proporcionaram ao controlador maior flexibilidade e capacidade de processamento, isto significou aumento na capacidade de memória e de entradas/saídas, permitiu entradas/saídas remotas, controle analógico, controle de posicionamento, comunicações, etc. A expansão de memória permitiu um programa de aplicação maior e uma maior quantidade de dados de forma que os programas de controle não ficassem restritos à lógica e seqüenciamento, mas também realizassem aquisição e manipulação de dados. Com o desenvolvimento

do controle analógico, o controlador programável preencheu o "gap" entre controle discreto e controle contínuo.

Os custos com fiação foram reduzidos significativamente com a capacidade do controlador de comunicar-se com subsistemas de entrada/saída localizados em pontos remotos, distante da unidade central de processamento e perto do equipamento a ser controlado. Ao invés de trazer centenas de fios para o armário do CP, os sinais dos subsistemas podem ser multiplexados e transmitidos por um único par de fios trançados. Esta técnica permitiu a decomposição de grandes sistemas em pequenos subsistemas melhorando a confiabilidade, manutenção e partida gradual dos subsistemas principais.

Em 1979 foi desenvolvida a rede de comunicação de alta velocidade (Data Highways - no jargão dos fabricantes da época) permitindo um controle sincronizado entre vários controladores, comunicação com microcomputadores e outros sistemas situados em um nível funcional superior. Com isto foi possível combinar o desempenho do controlador programável com a capacidade de controle distribuído de alta velocidade e interface com computadores resultando em uma grande potencialidade de controle e supervisão.

Atualmente, existem vários tipos de controladores, desde pequena capacidade até os mais sofisticados realizando operações que antes eram consideradas específicas para computadores. A evolução do hardware, conduziu a melhoras significativas nas características do controlador, entre outras citamos:

- Redução no tempo de varredura;
- Interfaces de E/S microprocessadas. Ex.: módulo PID, módulo ASCII, módulo de posicionamento;
- Uma Interface Homem Máquina (IHM) mais poderosa e amigável.

No software também surgiram novas características, tais como:

- Linguagem em blocos funcionais e estruturação de programa;
- Linguagens de programação de alto nível, baseadas em BASIC;
- Diagnósticos e detecção de falhas;
- Operações matemáticas em ponto flutuante através de coprocessadores matemáticos, etc.

3 COMPONENTES BÁSICOS

O Controlador Programável é um dispositivo de estado sólido usado para controlar máquinas ou processos por meio de um programa armazenado e realimentado por dispositivos de entrada e saída. A NEMA - National Electrical Manufacturers Association definiu, em 1978, um padrão para controladores programáveis como sendo "um aparelho eletrônico digital que usa uma memória programável para armazenamento interno de instruções para implementar funções específicas tais como lógica, seqüenciamento, temporização, contagem e operações aritméticas, para controlar máquinas ou processos através de módulos de entradas/saídas analógicos ou digitais". A Figura 1 ilustra o diagrama em blocos do controlador.

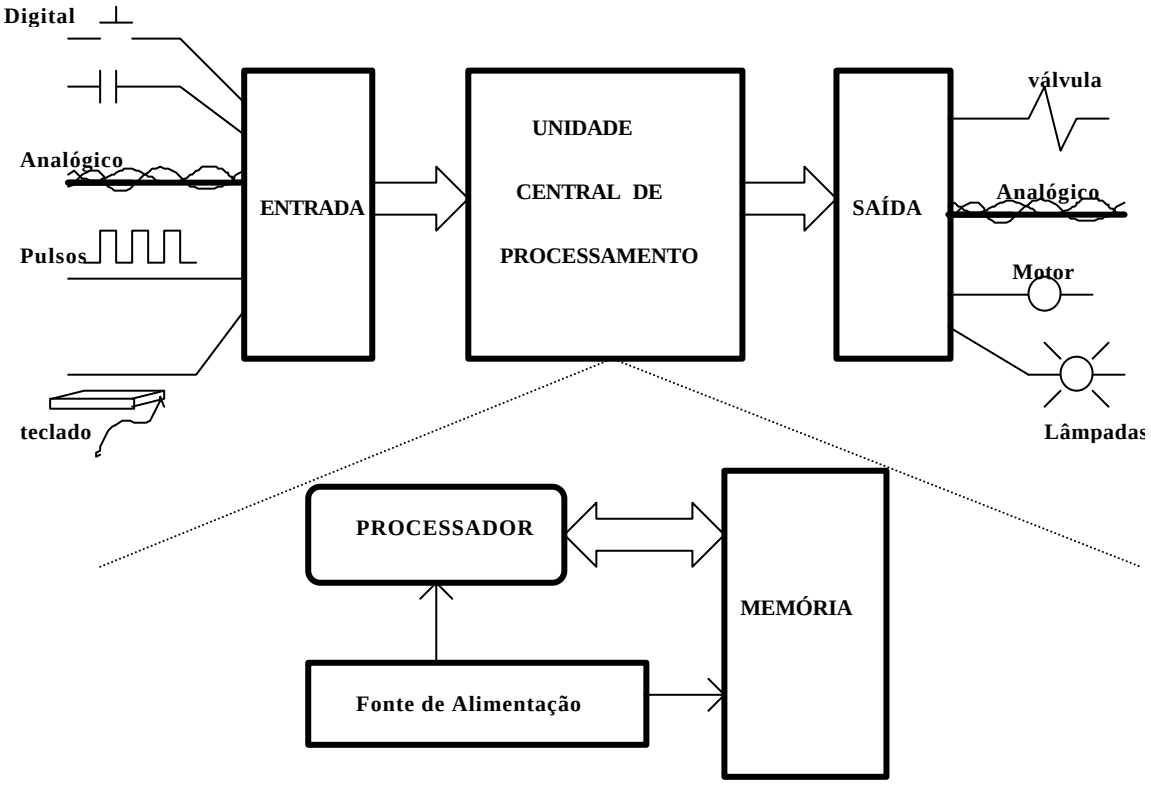


Figura 1 - Componentes básicos do CP

Um controlador programável, independente do tamanho, custo ou complexidade, consiste de cinco elementos básicos:

- Processador;
- Memória;

- Sistema de entradas/saídas;
- Fonte de alimentação;
- Terminal de programação.

A três partes principais (processador, memória e fonte de alimentação) formam o que chamamos de CPU - Unidade Central de Processamento.

O Processador lê dados de entrada de vários dispositivos, executa o programa do usuário armazenado na memória e envia dados de saída para comandar os dispositivos de controle. Este processo de leitura das entradas, execução do programa e controle das saída é feito de uma forma contínua e é chamado de ciclo de varredura.

O sistema de entrada/saída forma a interface pelo qual os dispositivos de campo são conectados ao controlador. O propósito desta interface é condicionar os vários sinais recebidos ou enviados ao mundo externo. Sinais provenientes de sensores tais como push-buttons, chaves limites, sensores analógicos, chaves seletoras e chaves tipo tambor (thumbwheel), são conectados aos terminais dos módulos de entrada. Dispositivos que devem ser controlados, como válvulas solenóides, lâmpadas pilotos e outros, sãos conectados aos terminais dos módulos de saída.

A fonte de alimentação fornece todas as tensões necessárias para a devida operação do CP e da interface dos módulos de entrada e saída.

Dependendo de como estas partes estão fisicamente organizadas podemos ter dois tipos de estrutura. A primeira é do tipo compacta, onde todos os componentes são colocados em uma única estrutura física, isto é, o processador, a memória, a fonte e o sistema de entrada/saída são colocados em um gabinete ficando o usuário com acesso somente aos conectores do sistema E/S. Este tipo de estrutura é normalmente empregada para CP´s de pequeno porte.

A segunda estrutura apresenta um abordagem modular onde cada componente ou um conjunto deles é colocado em um módulo. Podemos ter processador e memória em um único módulo com fonte separada ou então estas três partes juntas em um único gabinete. O sistema de entrada/saída é decomposto em módulos de acordo com suas características. Estes módulos são então colocados em racks formando uma configuração de médio e grande porte. A Figura 2 ilustra as estruturas descritas.

Outro componente de controlador programável é o dispositivo de programação. Embora seja considerado como parte do controlador, o terminal de programação, como era chamado antes, é requerido apenas para entrar com o programa de aplicação na memória do controlador. Uma vez carregado o programa o terminal pode ser desconectado do controlador. Atualmente se usa o microcomputador para programar o CP e devido à capacidade de processamento do mesmo, este também é utilizado para monitoração e depuração do programa.

Figura 2 – Estruturas compacta e modular (Cortesia GE-Fanuc)

4 CONTROLADOR PROGRAMÁVEL VERSUS PAINEL DE RELÉS

Controladores Programáveis ou painéis de relés? Esta foi provavelmente uma pergunta muito comum entre os engenheiros de sistemas, controle, projetistas, etc. Não se pode generalizar, mas é certo que alta qualidade e produtividade não podem ser obtidas, de maneira econômica, sem equipamento de controle eletrônico. Com o rápido desenvolvimento e crescimento da competição, o custo do controlador programável tem caído significativamente a ponto de que o estudo de CP versus relés, no ponto de vista de custo não ser mais válido. As aplicações com controladores programáveis podem, agora, serem avaliadas por seus próprios méritos. Requisitos tais como indicados abaixo seguramente levam à opção pelo CP ao invés de relés:

- Necessidade de flexibilidade de mudanças na lógica de controle;
- Necessidade de alta confiabilidade;
- Espaço físico disponível pequeno;
- Expansão de entradas e saídas;
- Modificação rápida;
- Lógicas similares em várias máquinas;
- Comunicação com computadores em níveis superiores.

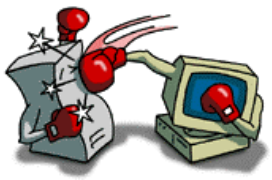
Embora o sistema eletromecânico, em pequenas e até médias aplicações, possa apresentar um custo inicial menor, esta vantagem poderá ser perdida considerando-se a relação custo/benefício que o CP proporciona.

A Figura 3 ilustra uma comparação entre o quadro de relés e o quadro de CP's. Pode ser observado que a implementação da lógica através de relés dificulta a manutenção e torna o sistema menos flexível à mudanças. A lógica é realizada por fios e qualquer modificação na lógica exige uma conexão adequada dos fios, envolvendo operações com os contatos NA e NF dos relés.

Figura 3 – Comparação entre os quadros de relés e CP’s

5 COMPUTADOR INDUSTRIAL VERSUS CONTROLADOR PROGRAMÁVEL

A arquitetura de um controlador programável é basicamente a mesma que um computador de propósito geral. Entretanto existem algumas características importantes que diferem o CP dos computadores. Podemos dizer que todos os CP´s são computadores por definição, mas nem todos os computadores são CP´s. A diferença está nos métodos de programação, operação, considerações ambientais e manutenção. A Figura 4 ilustra uma comparação entre computadores industriais e CP onde podem ser vistos os pontos fortes e os pontos fracos dos computadores industriais.



Pontos fortes

Interface Gráfica
Tempo de Programação
Não utilizar Hardware Proprietário
Arquitetura Aberta
Rede de comunicação TCP/IP
Simulação do Programa
Várias Linguagens de Programação
Comunicação com Supervisorio
Utilização de vários Hardwares de E/O
Facilidade de efetuar calculos complexos

Pontos fracos

Confiabilidade do Sistema Operacional
Confiabilidade do Microcomputador
Velocidade de Atualização de E/O (Rack)
Eventuais Bugs de Software

Figura 4 – Comparação do Microcomputador Industrial com CP’s

Os CP´s foram especificamente projetados para operar em ambientes industriais. Um CP pode operar em áreas com quantidades substanciais de ruídos elétricos, interferências eletromagnéticas, vibrações mecânicas, temperaturas elevadas e condições de umidade adversas. Uma especificação típica de CP inclui temperaturas na faixa de 0 a 60 ^{o}C e umidade relativa de 5 a 95 %.

A segunda distinção dos CP´s é que o hardware e o software foram projetados para serem operados por técnicos não especializados (nível exigido para a manutenção e operação de computadores). Usualmente, a manutenção é feita pela simples troca de módulos e existem softwares que auxiliam na localização de defeitos. As interfaces de hardware para conexão dos dispositivos de campo estão prontas para uso e são facilmente intercambiáveis (estrutura modular). A programação é geralmente feita em uma linguagem parecida com os diagramas de relés.

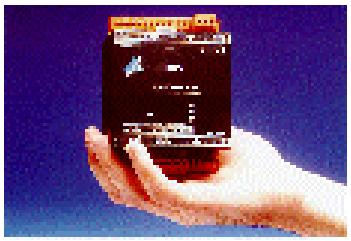
O software residente, desenvolvido pelo fabricante, e que determina o modo de funcionamento do controlador também caracteriza uma diferença fundamental. Este software realiza funções de acesso ao hardware, diagnósticos, comunicações e determina o funcionamento do controlador em um modo de operação dedicado (ciclo de varredura) e totalmente transparente ao usuário.

6 CPU E CICLO DE VARREDURA

A CPU é o elemento responsável pelo gerenciamento e processamento das informações do sistema. Em uma análise mais detalhada podemos concluir que a CPU é na verdade um microprocessador conectado a circuitos auxiliares, tais como, memórias, circuitos de temporização e interface, etc. Este microprocessador expressa a complexidade e a capacidade do controlador. Vejamos por exemplo, um controlador que use a família Intel 80X86. Esta família representa a base dos microcomputadores pessoais do tipo IBM/PC, assim sendo, pode-se esperar que este controlador apresente características que o habilite a ser aplicado em operações complexas ou de grande porte. Uma observação nestes controladores é que além da sua capacidade de processamento pode-se adicionar outros microprocessadores destinados a realizar cálculos matemáticos ou booleanos, são os chamados coprocessadores. Neste ponto observamos a tendência em transformar um Controlador Programável em um Computador Industrial, para aplicações em processos contínuos. A Figura 5 ilustra CPU’s de vários fabricantes.

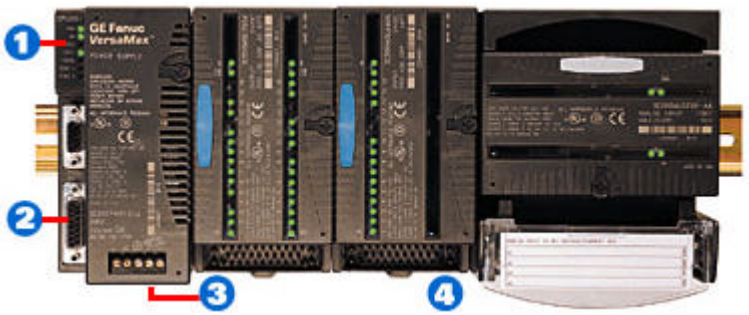
Atos – Micro CLP

Atos – Série Modular



Altus - Série Piccolo

Altus - Série Quark



GE-FANUC - Versamax

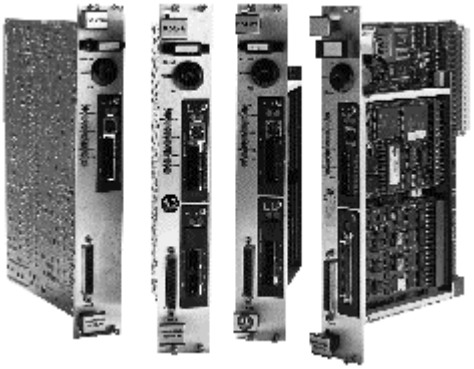
GE-FANUC – Série 9030

GE-FANUC – Série 9070

Rockwell

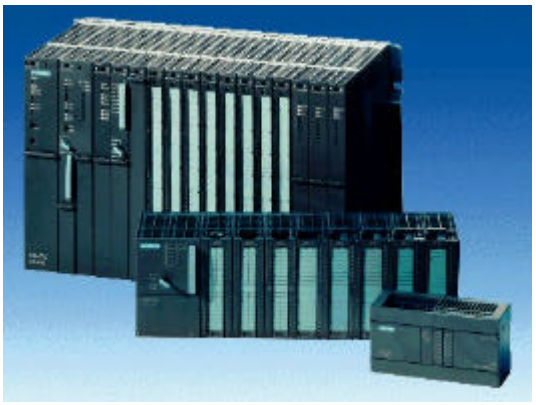
Rockwell - MicroLogix

Rockwell – PLC 5

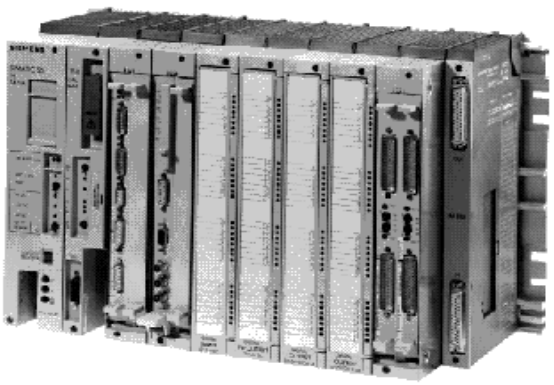


Rockwell – PLC 5 VME Bus

Siemens - Série S5 – 95U



Siemens - Série S7



Série S5 – 115U

Figura 5 – CPU’s de vários fabricantes

A CPU coordena as atividades do sistema, interpretando e executando um conjunto de instruções, conhecido como programa Executivo ou Monitor. Este programa realiza um papel similar ao sistema operacional de um microcomputador, com a diferença de ser exclusivamente para controle e monitoração do CP. O Executivo se encontra armazenado em memórias não voláteis e é considerado como parte do sistema.

Todas as funções relacionadas com a operação do controlador estão definidas no programa Executivo. Existem funções básicas que são encontradas em qualquer controlador e outras funções que são consideradas especiais e constituem o diferencial entre controladores de linhas ou fabricantes diferentes. Entre as funções básicas encontram-se:

- Diagnósticos: watch-dog, bateria, checksum;
- Modo de operação: em execução (run) e parado (stop);
- Comunicação: implementação de diversos tipos de protocolos.

De uma forma geral, podemos visualizar estas funções no frontal do controlador através de LED´s de sinalização que indicam o estado operacional do equipamento. Estas funções normalmente são encontradas independentemente da arquitetura física do controlador, isto é, se em forma modular ou compacta. A Figura 6 ilustra um diagrama típico de um processador, mostrando as funções e conectores para conexão de dispositivos de programação, E/S e energização do CP.

O estado operacional do controlador pode ser definido através de chaves no próprio frontal (não ilustrado na figura) ou através do aparelho programador. Por exemplo, pode-se colocar o CP em modo de execução (LED "RUN" aceso) através de um comando do programador, e uma vez neste estado o CP executará o programa de usuário sob o comando do programa Executivo. Por outro lado, pode-se colocar o CP no modo de programação (LED "PROG" aceso), o que habilita o controlador a receber o programa do usuário. Os outros LED´s de sinalização indicam a potência (PWR), comunicação ativa (COM) e bateria baixa (BAT).

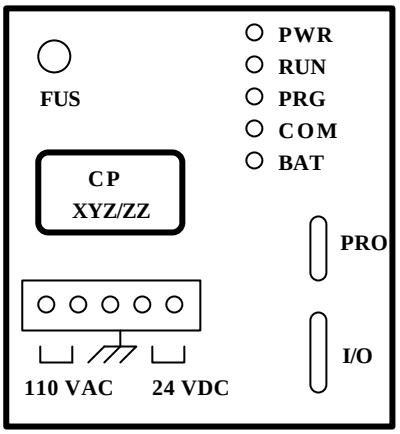


Figura 6 – Funções típicas no frontal da CPU

6.1 PRINCÍPIO DE FUNCIONAMENTO

O Controlador Programável tem uma forma particular de trabalhar que caracteriza o seu funcionamento. O controlador opera executando uma seqüência de atividades definidas e controladas pelo programa Executivo. Este modo de operação ocorre em um ciclo, chamado de Ciclo de Varredura ("Scan"), que consiste em :

- leitura das entradas externas;
- execução da lógica programada;
- atualização das saídas externas.

Na fase de leitura das entradas, o Processador endereça o sistema de E/S, obtém os estados dos dispositivos que estão conectados, e armazena estas informações na forma de bits "1" ou "0", dependendo do estado obtido (ponto energizado eqüivale ao binário "1" e ponto desenergizado ao binário "0"). A região da memória utilizada para armazenar estas informações é chamada de Tabela Imagem das Entradas - TIE.

Na fase de execução da lógica programada pelo usuário, a CPU consulta a TIE para obter os estados dos dispositivos. Nesta fase, os resultados das lógicas programadas cujas saídas tenham um ponto correspondente no rack de saída são armazenados em uma área de memória que é chamada de Tabela Imagem das Saídas - TIS. As lógicas que possuem saídas internas serão armazenadas na área correspondente. Durante a execução da lógica programada, se for necessário a referência a uma saída qualquer, dentro do mesmo ciclo, esta tabela é consultada. Observe que durante esta fase não é feita

nenhuma referência a pontos externos (entrada ou saída), a CPU opera com informações obtidas da memória.

Na fase de atualização de saídas, a CPU executa uma varredura na tabela TIS e atualiza as saídas externas, endereçando o Sistema de E/S para atualizar o estado dos dispositivos externos de acordo com o resultado da lógica programada. A seguir, o ciclo é reiniciado e a operação continua enquanto se mantém o controlador no modo de execução ("Run").

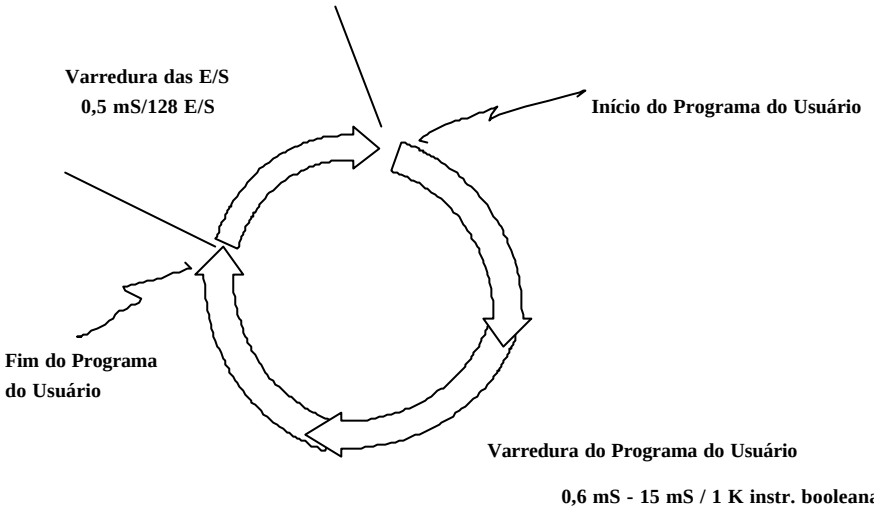


Figura 7 – Ciclo de varredura

O tempo necessário para a varredura varia de controlador para controlador e depende de muitos fatores (tamanho da palavra, clock, instruções programadas, etc.). O fabricante especifica este tempo baseado na quantidade de instruções, normalmente instruções booleanas, e quantidade de entradas/saídas. Qualquer outra função programada aumenta este tempo de varredura.

Este processo de varredura pode ser inadequado para entradas rápidas, isto é, entradas com freqüência acima de 10 Hz. Neste caso devemos utilizar de funções especiais do CP para interromper a varredura do programa e atualizar o estado de uma entrada ou de uma saída imediatamente. Este processo é realizado por software e também está limitado à execução do programa do usuário. Em aplicações de alta velocidade, tais como em sensores eletrônicos por pulsos, é aconselhável o uso de módulos específicos (contadores de alta velocidade).

A interrupção do ciclo de varredura para atualização pode ocorrer de duas maneiras:

- Interrupção para entrada imediata: o ciclo é interrompido para uma leitura de módulos de

entrada. Após a leitura ocorre a atualização da Tabela Imagem das Entradas com os pontos selecionados e o programa prossegue normalmente;

- Interrupção para saída imediata: após a execução de uma lógica pode ser necessário atualizar imediatamente as saídas externas. Neste caso, programa-se uma Instrução de Saída Imediata para atualizar o estado externo. Observe que a CPU acessa a Tabela Imagem de Saída, que já possui os resultados correntes e escreve no endereço do módulo de saída referenciado na instrução.

Além das duas maneiras mais usuais apresentadas acima, a varredura normal do programa de usuário pode ser alterada por uma entrada especial que, tendo sofrido uma variação no seu estado, gera uma interrupção na CPU. Esta interrupção desvia a execução do programa para uma subrotina especial que pode ou não ser programada pelo usuário.

O tempo de varredura é uma consideração importante na seleção do controlador. Este indica a rapidez com que o controlador pode reagir às entradas de campo e resolver corretamente a lógica de controle. Por exemplo, se um controlador tem um tempo de varredura de 50 ms e necessita monitorar um sinal de entrada que pode mudar de estado a cada 20 ms, o controlador nunca será capaz de aquisitar este sinal, resultando em um mau funcionamento da aplicação.

6.2 ORGANIZAÇÃO DA MEMÓRIA

Antes de apresentar a organização de memória do CP é preciso entender a estrutura básica da memória.

6.2.1 Estrutura da memória

A memória do controlador programável pode ser visualizada como um grande conjunto bidimensional de células unitárias de armazenamento, cada um das quais armazenam uma única informação na forma de “1” ou “0”. É óbvio, portanto, que o sistema de numeração binário seja usado para representar a informação armazenada na memória. Como BIT é o anacronismo para BInário digiT e cada célula pode armazenar um bit, cada célula é chamada de bit. Um bit é então a menor unidade de estrutura de memória e armazena informações na forma de 1s e Os. O bit é considerado ON se a informação armazenada é 1 e OFF se a informação armazenada é zero. Portanto, um bit é suficiente

para armazenar o estado de chaves, botoeiras, fim de cursos, motores e outros dispositivos externos que podem ser conectados ao CP.

Frequentemente é necessário que o CP manipule mais do que um bit. Por exemplo, é muito mais eficiente manipular um grupo de bits quando se deseja transferir dados para/da memória. Um grupo de 8 bits manipulado simultaneamente é chamado de byte e um grupo de 16 bits é chamado de palavra. A Figura 8 abaixo ilustra os conceitos definidos acima.

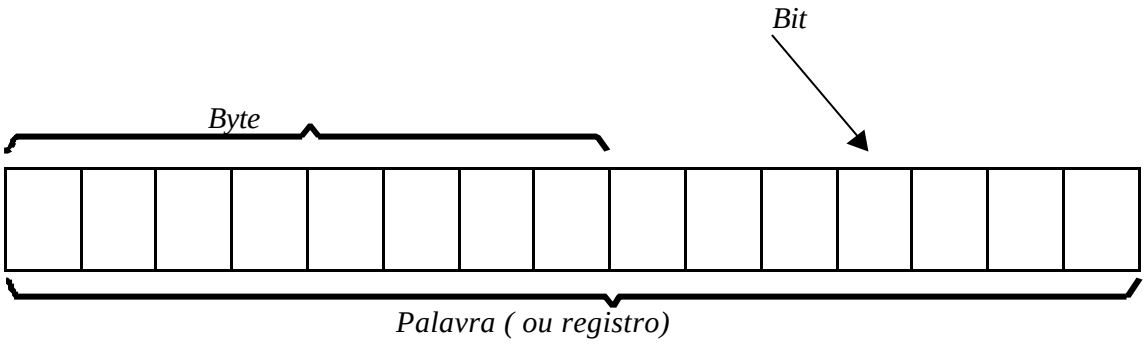


Figura 8 – Estrutura da memória

6.2.2 Organização da memória

Basicamente, o sistema de memória pode ser dividido em três partes: memória do sistema, tabela de dados e memória da aplicação.

A **memória do sistema** é destinada ao armazenamento do programa executivo responsável por toda a operação de funcionamento do controlador programável. O programa executivo é um conjunto de programas armazenado permanentemente na memória do controlador com o objetivo de controlar e supervisionar as atividades do sistema, tais como: controle do ciclo de varredura, comunicação como os dispositivos periféricos, diagnósticos e outras atividades. Normalmente se refere à memória do sistema como “firmware”, para expressar o conjunto software e hardware necessário para o CP funcionar corretamente.

A **tabela de dados** é utilizada para armazenar qualquer dado associado com o controle do sistema, tais como: estados das entradas e saídas conectadas ao controlador programável, estados internos, valores preset de contadores e temporizadores. A tabela de dados é importante e será tratada no próximo item.

A **memória da aplicação** é destinada ao armazenamento da lógica de controle definida pelo usuário, isto é, do programa de aplicação ou programa do usuário.

6.2.3 Tabela de dados

A tabela de dados define o endereçamento dos diversos tipos de dados que um controlador pode manipular. A Figura 9 ilustra a divisão da tabela de dados.

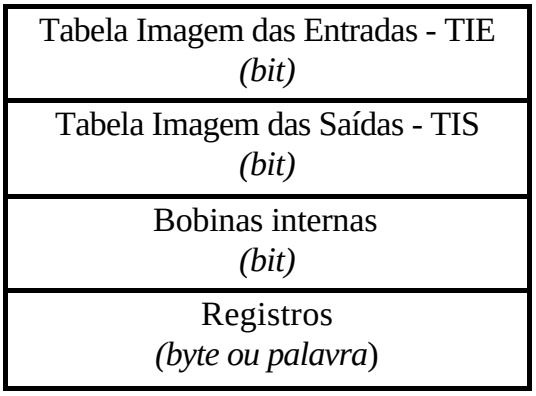


Figura 9 – Mapa de memória da tabela de dados

A estrutura da memória pode ser de dois tipos:

- Estado: Informações do tipo ON/OFF representado por 1s e 0s, conforme ilustrado para imagem das entradas e saídas e para as bobinas internas. Bobinas internas representam saídas que não estão disponíveis externamente no controlador programável e portanto não podem acionar dispositivos conectados ao CP.
- Número ou códigos: Informações representadas por um grupo de bits (byte ou palavra).

Tabela Imagem das entradas - TIE

A tabela imagem das entradas armazena o estado das entradas digitais conectadas ao CP. Isto significa que para cada entrada digital tem um bit correspondente na tabela imagem das entradas. Se a entrada estiver energizada (ON), o bit correspondente na tabela imagem é 1. Se a entrada estiver desenergizada (OFF), o bit correspondente na tabela imagem é 0. Veja a Figura 10. Durante o início do ciclo de varredura a tabela imagem das entradas é atualizada para refletir o estado corrente do dispositivo.

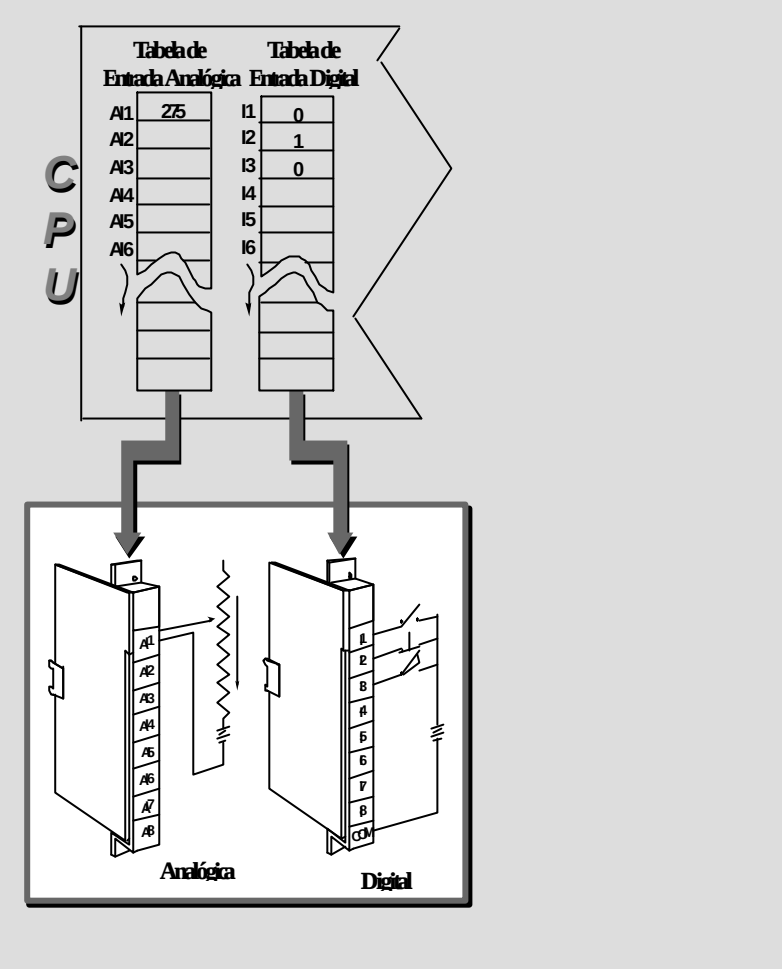


Figura 10 – Tabela Imagem das Entradas

Tabela Imagem das saídas - TIS

A tabela imagem das saídas armazena o estado das saídas externas conectadas ao CP. Durante a execução do programa do usuário, quando o processador interpreta e executa a lógica programada, esta tabela imagem é atualizada. De forma similar, para cada ponto de saída externo ao CP deve existir um ponto correspondente nesta tabela imagem. Veja a Figura 11.

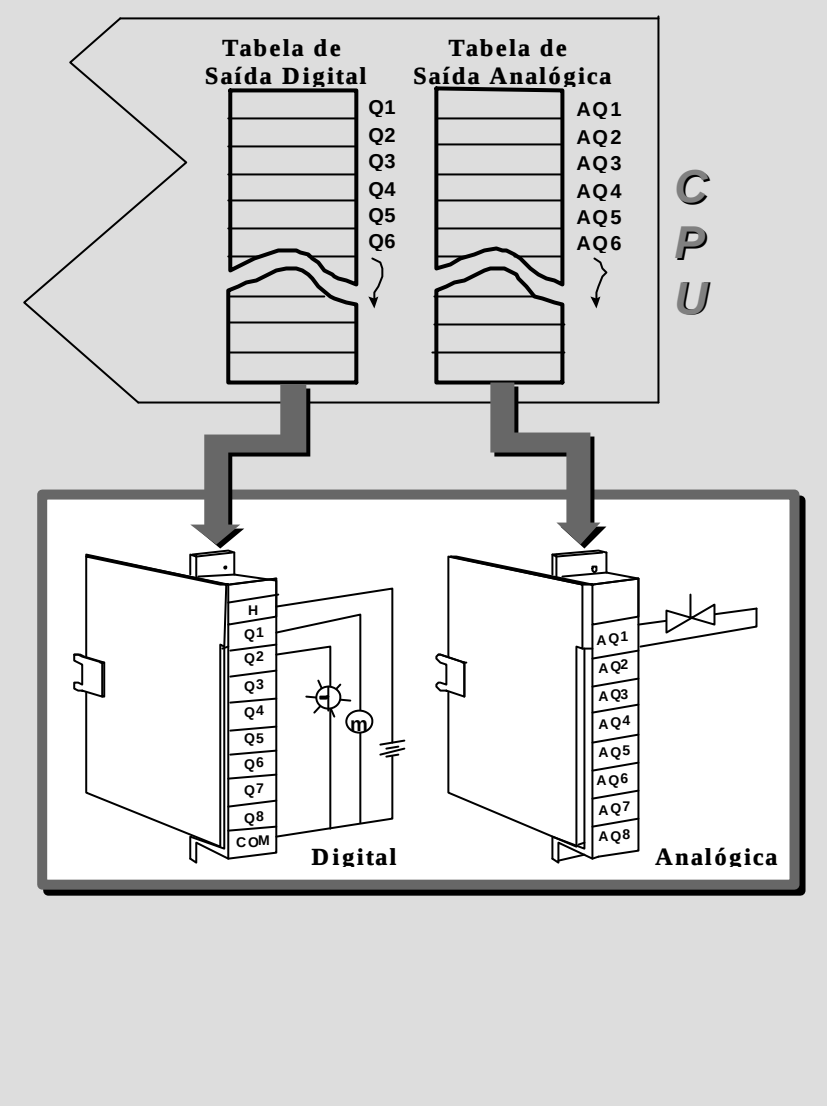


Figura 11 – Tabela Imagem das Saídas

Bobinas internas

Bobinas internas, também chamadas de bobinas lógicas ou saídas internas, são utilizadas somente para realizar intertravamentos e armazenamentos de estados lógicos internos no programa do usuário. Diferentemente das saídas externas, as bobinas internas não possuem um ponto físico correspondente no sistema de saídas do CP.

Registros

Os registros são posições de memórias destinadas a armazenar informações quantitativas. Podem ser utilizados para armazenar valores preset de contadores e temporizadores bem como qualquer dado numérico manipulado pelo CP.

6.2.4 Memória da aplicação

A memória da aplicação é uma região com características de escrita e leitura aleatória. Esta memória é destinada a armazenar o programa do usuário. O programa do usuário contém a lógica de controle através do dispositivo de programação (geralmente um microcomputador ou notebook) e descarregada na memória do CLP através do software de programação específico do fabricante.

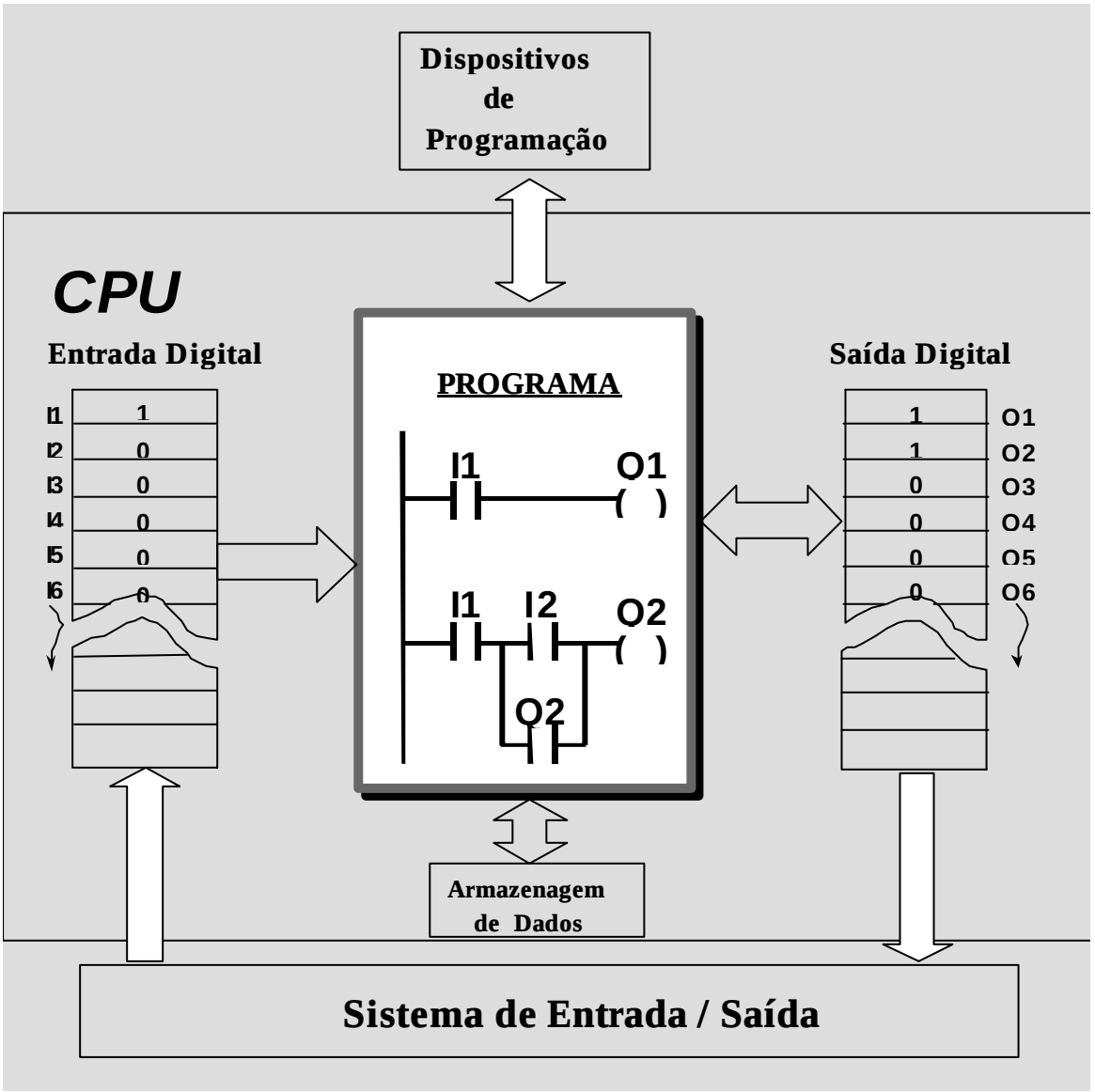


Figura 12 – Programa do usuário

7 SISTEMA DE ENTRADAS E SAÍDAS

O Sistema de Entradas/Saídas fornece a conexão física entre o mundo externo (equipamentos de campo) e a unidade central de processamento. Através de circuitos de interface, o controlador pode sensorar ou medir quantidades físicas, independente da máquina ou processo, tais como: proximidade, posição, movimento, nível, temperatura, pressão, corrente e tensão. Baseado no estado sensorado ou no valor medido, e nas instruções do programa de usuário, o processador comanda os dispositivos de controle conectados ao subsistema de saída. Estes dispositivos podem ser válvulas, motores, bombas, alarmes, etc.

Os predecessores dos atuais controladores programáveis eram limitados a interfaces de entradas e saídas discretas que só permitiam a conexão de dispositivos tipo ON/OFF. Estas limitações permitiam ao controlador apenas um controle parcial em muitas aplicações.

Atualmente os controladores possuem uma grande variedade de interfaces (analógicas e discretas) o que permite sua aplicação em praticamente qualquer tipo de controle.

7.1 ENTRADAS E SAÍDAS DISCRETAS

A classe mais comum de interface de entrada/saída é o tipo discreto. Esta interface conecta dispositivos de entrada de campo, que fornecem sinais de entrada independentes e distintos em natureza dos sinais eletrônicos da interface, ou dispositivos de saída para campo que necessitem de sinais independentes e distintos em natureza dos sinais eletrônicos da interface para controlar seu estado. Estas características limitam a interface em sensorar sinais do tipo ON/OFF ou fechado/aberto. Para esta interface o sinal de entrada é essencialmente uma chave que esta aberta ou fechada. Da mesma forma, o controle da saída é limitado a dispositivos que somente requerem comutação em dois estados, tais como ON/OFF ou ligado/desligado.

Tabela 1 – Entradas e saídas discretas

Dispositivos de Entrada	Dispositivos de Saídas
Chaves seletoras	Alarmes
Push buttons	Relés de controle
Fotoelétricos	Ventiladores
Chaves limites	Lâmpadas
Contatos de relés	Buzinas
Chaves de nível	Motores
Chaves de proximidade	Válvulas solenóides

Cada entrada ou saída é alimentada por alguma fonte de alimentação que podem ser ou não de mesma magnitude ou tipo (p. ex. 120 VAC, 24 VDC). Por esta razão, circuitos de interface são disponíveis para vários valores de tensão AC e DC, como listado na Tabela 2.

Quando em operação, se a chave de entrada é fechada, a interface de entrada monitora a tensão fornecida e a converte em um sinal aceitável para o processador indicando o estado do dispositivo. Um estado lógico 1 indica um estado ON ou fechado do dispositivo externo e um estado lógico 0 indica um estado OFF ou aberto do dispositivo.

Tabela 2 - Valores padrões para interfaces discretas

Entradas	Saídas
12/48 VAC	12/48 VAC
12/48 VDC	12/48 VDC
110/220 VAC	110/220 VAC

Nível TTL	Contato de relé

O circuito de entrada é composto por duas seções principais potência e lógica. Estas duas seções são normalmente desacopladas eletricamente por um circuito isolador (ver a Figura 13).

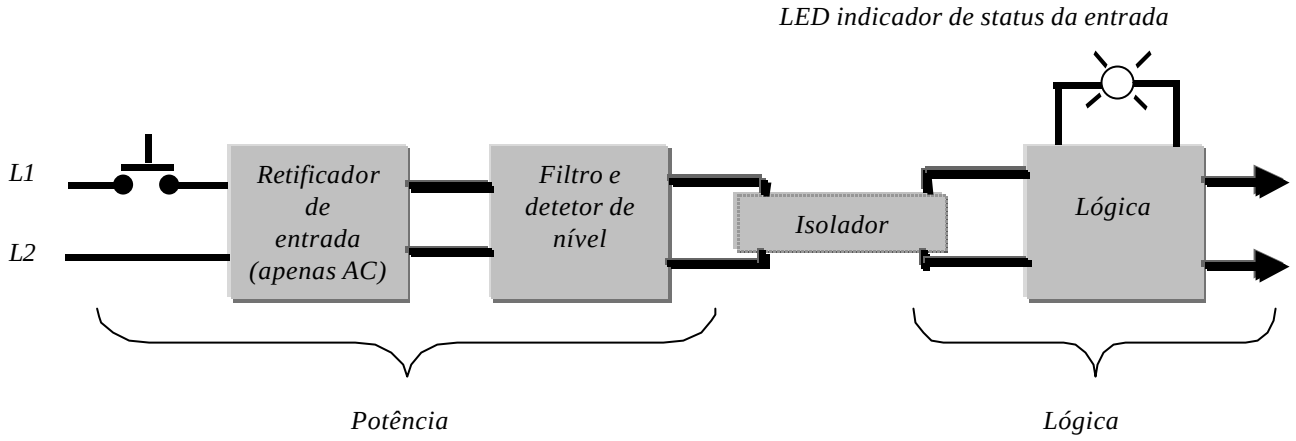


Figura 13 – Diagrama em blocos típico de uma entrada discreta

A seção de potência basicamente realiza a função de conversão da tensão de entrada (110 VAC, 220 VAC, Tc) para um nível DC compatível com a interface. Após o retificador, o sinal DC passa por um circuito filtro que elimina o ruído elétrico e realiza o antibounce do sinal de entrada. Este filtro provoca um atraso de 9-25 ms. O circuito threshold detecta quando o sinal atinge o nível de tensão especificado para o nível lógico. Se o sinal excede e permanece acima do limite de tensão por um tempo no mínimo igual ao atraso do filtro, o sinal é reconhecido como uma entrada válida.

Quando um sinal válido é detectado, o circuito isolador gera um sinal na seção lógica completando assim uma transição eletricamente isolada de um sinal AC para o nível lógico correspondente. O sinal DC na seção lógica fica disponível para o processador através do seu barramento de dados.

A conexão típica para os dispositivos de campo é mostrada na Figura 14. A maioria dos cartões de entrada utiliza um indicador LED ou Neon para indicar a presença do sinal de entrada (potência). Se o indicador estiver aceso a chave correspondente deve estar fechada. Um indicador LED pode estar

disponível para indicar o estado lógico 1 na seção lógica. Desta forma, visualmente pode ser detectado o funcionamento correto do canal de entrada.

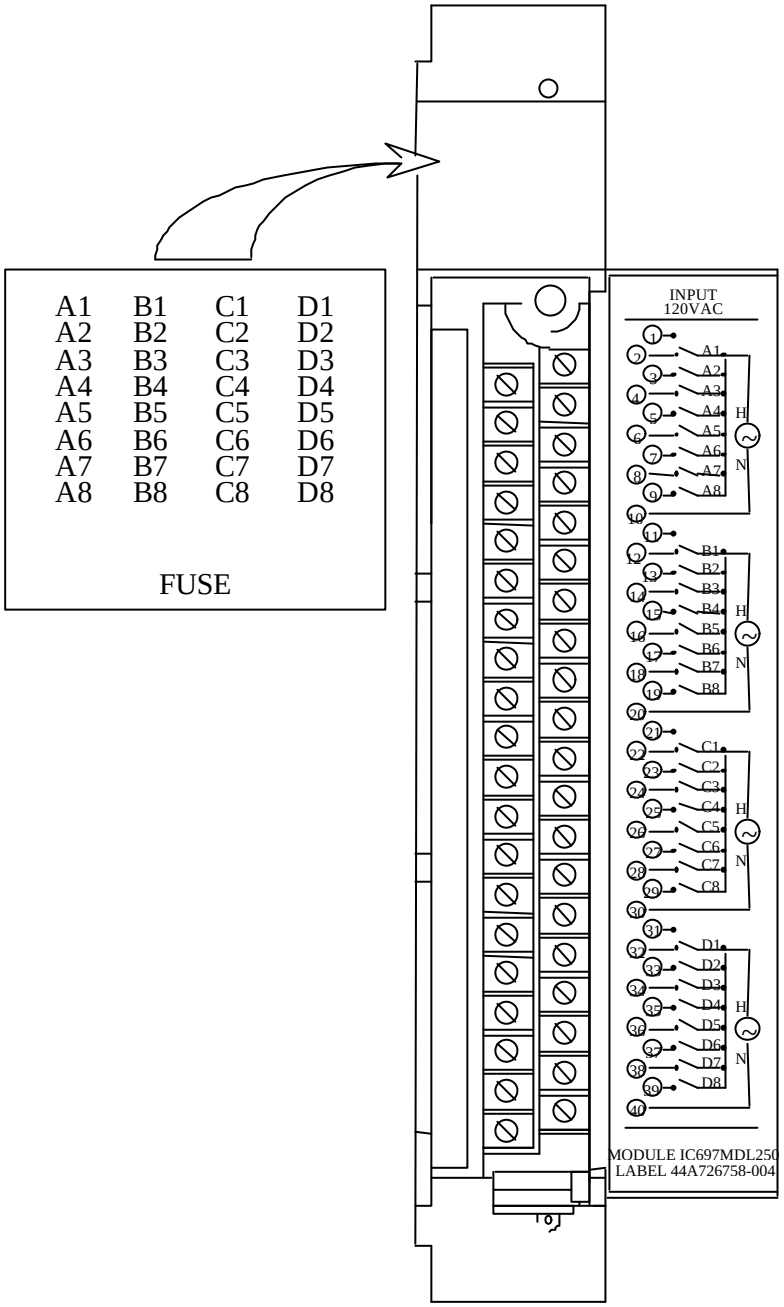


Figura 14 - Diagrama de conexões típicas para módulo de entradas da GE-Fanuc

O módulo de entrada para AC/DC possui todos os circuitos necessários para a interface entre os dispositivos ON/OFF, tais como, chaves seletoras, limitadores e os níveis lógicos exigidos pelo controlador programável. Todos os módulos de entrada contém os circuitos para dois canais de entrada individualmente isolados. Cada canal aceita um sinal AC (48-63 Hz) ou um sinal de entrada

DC de tensão especificada. A lógica é 1 quando o sinal de entrada atinge um limite pré determinado. A isolação elétrica entre o sinal de entrada e o sinal lógico é obtida através de opto-acopladores.

De forma similar, um circuito de saída discreto é composto por duas seções principais acopladas por um circuito isolador (ver Figura 15). Durante uma operação normal, o processador envia para o circuito lógico o estado da saída de acordo com a lógica do programa. Se a saída é energizada, o sinal lógico 1 proveniente do processador alimenta a seção lógica de forma a energizar o dispositivo de campo.

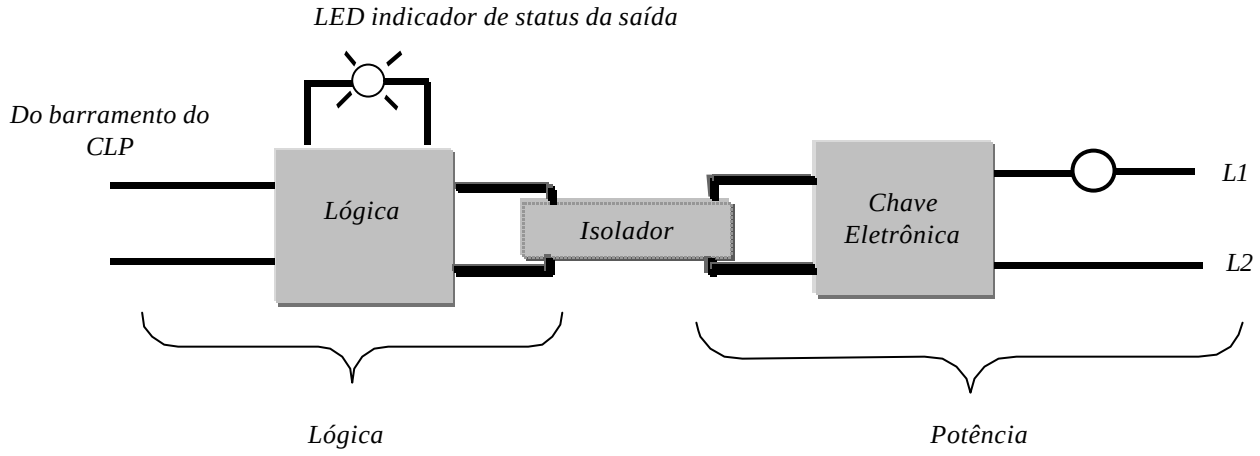


Figura 15 – Diagrama em blocos típico de uma saída discreta

Em um módulo de saída AC, a seção de potência geralmente usa um Triac ou um SCR para comutar a carga. A chave AC é normalmente protegida por um circuito RC (snubber) e freqüentemente um MOV - Metal Oxide Varistor, que são utilizados para limitar o pico de tensão a um valor seguro e também para evitar que ruídos elétricos perturbem a operação do circuito. Um fusível pode ser fornecido no circuito de saída para evitar que uma corrente excessiva danifique o elemento comutador. São fornecidos indicadores para sinalizar o estado lógico e o circuito de potência. Na presença de fusíveis o seu estado é sinalizado por indicadores.

O circuito de saída DC tem uma operação funcional similar à saída AC, entretanto, o circuito de potência geralmente emprega um transistor de potência para chavear a carga.

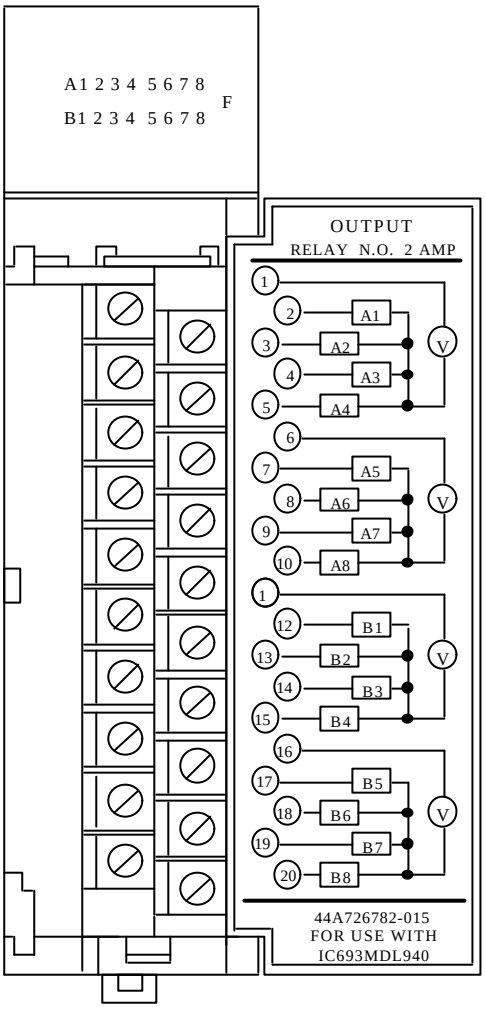


Figura 16 - Conexões entre módulo de saída discreta e dispositivos de campo

Os módulos de saída a contato de relé permitem que dispositivos de campo sejam comutados por contatos NA e NF de relés. Os contatos podem ser usados para comutar cargas AC ou DC, mas normalmente são utilizados em aplicações tais como:

- Multiplexação de sinais analógicos;
- Comutação de pequenas correntes a baixa tensões;
- Interface para controle de diferentes níveis de tensão.

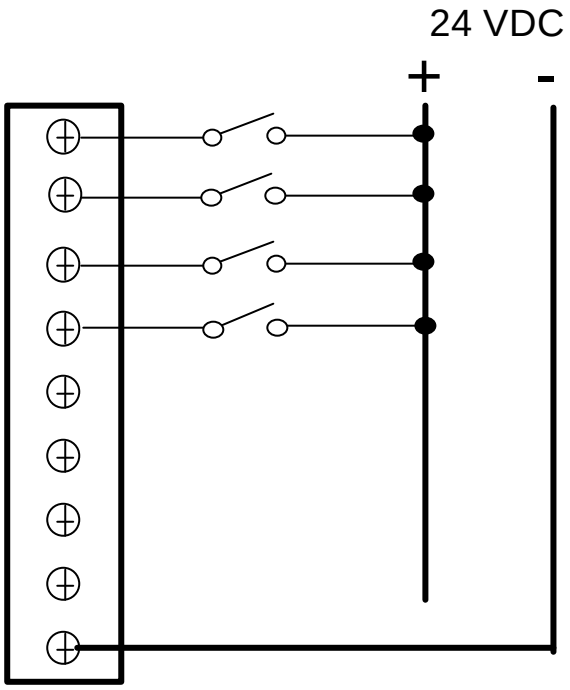
7.1.1 Lógica positiva e Lógica negativa

Os módulos de entrada e saídas DC podem ser classificados como Lógica Positiva ou Lógica Negativa, cujas características são descritas a seguir.

Entradas Discretas DC

- Lógica Positiva – o dispositivo externo aplica o pólo positivo da fonte na entrada digital. Também chamada de entrada tipo P
- Lógica negativa – o dispositivo externo aplica o pólo negativo da fonte na entrada digital. Também chamada de entrada tipo N.

A Figura 17 ilustra as conexões físicas para a entrada DC. Na lógica positiva, o dispositivo externo é conectado entre o potencial positivo da fonte e o terminal de entrada do módulo, assim o módulo de entrada absorve corrente do dispositivo externo. Na lógica negativa, o dispositivo externo é conectado entre o potencial negativo e o terminal de entrada do módulo, assim o módulo de entrada fornece corrente para o dispositivo externo.



Entrada Lógica Positiva (tipo P)

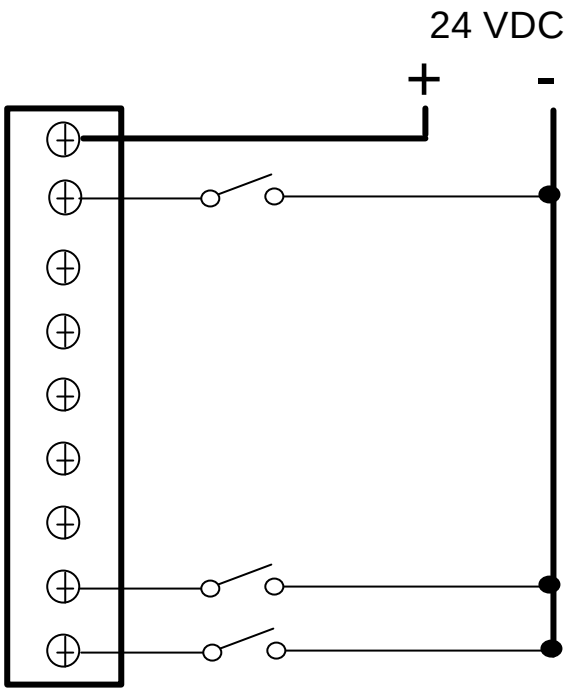
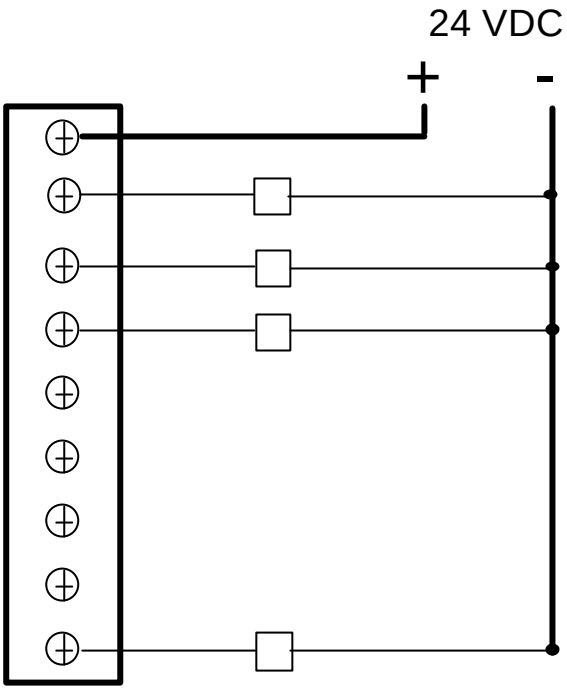


Figura 17 – Entrada Discreta DC: lógica positiva e negativa

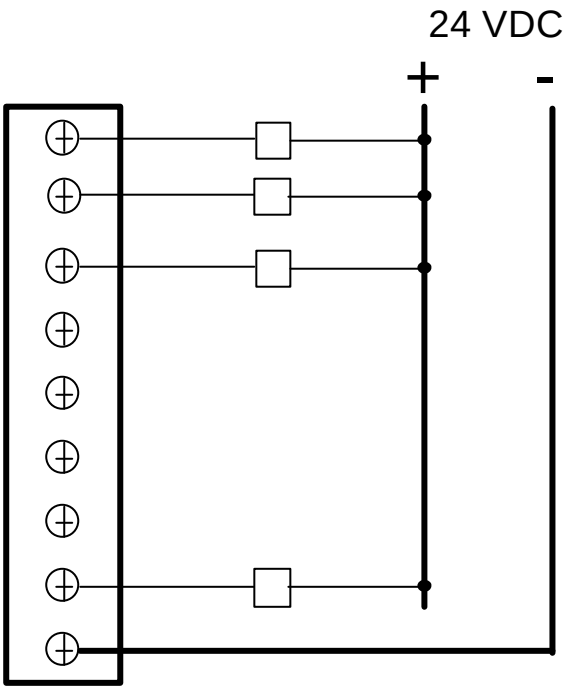
Saídas Discretas DC

- Lógica positiva – a carga recebe o pólo positivo da fonte de alimentação
- Lógica negativa – a carga recebe o pólo negativo da fonte de alimentação

A Figura 18 ilustra as conexões físicas para a saída discreta DC. Na lógica positiva, a carga é conectada entre o potencial negativo da fonte e o terminal de saída do módulo, assim o módulo de saída fornece corrente para a carga. Na lógica negativa, a carga é conectada entre o potencial positivo e o terminal de saída do módulo, assim o módulo de entrada consome corrente da carga.



Entrada Lógica Positiva (tipo P)



Entrada Lógica Negativa (tipo N)

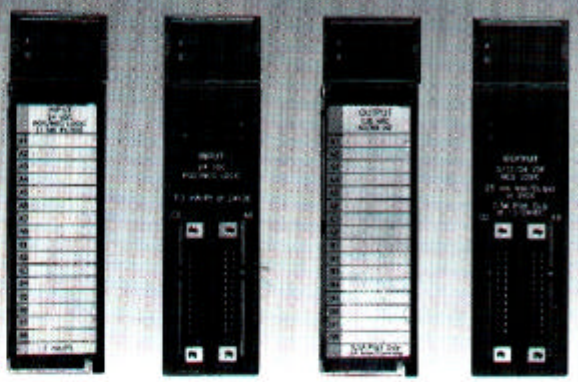
Figura 18 – Saída Discreta DC: lógica positiva e negativa

A Figura 19 ilustra exemplos de características de diversos módulos de entsrada e saídas discretas da GE-FANUC.

Discrete I/O Modules

A Wide Choice of Easy-to-Wire Modules

Input modules provide the interface between the PLC and external input devices such as proximity sensors, push buttons, switches, and BCD thumb-wheels. Output modules provide the interface between the PLC and external output devices such as contactors, interposing relays, BCD displays and indicator lamps. GE Fanuc offers a variety of modules that support different voltage ranges and types, current capacity, isolation, and response time to meet the needs of your application.



AC Voltage Input Module Specifications

Part Number	Description	Input Voltage	Number of Points	Response Times (ms) On	Response Times (ms) Off	Input Current	Trigger Voltage	Points per Common	Connector Type
IC693MDL230	120V Isolated	0-132	8	30	45	14.5 mA	74-132	1	TB*
IC693MDL231	240V Isolated	0-264	8	30	45	15 mA	148-264	1	TB
IC693MDL240	120V Input	0-132	16	30	45	12 mA	74-132	16	TB
IC693MDL241	24VAC/VDC	0-30	16	12	28	7 mA	11.5-30	16	TB
IC693MAR590	AC In/Relay Out	0-132	8	30	45	12 mA	74-132	8	TB

DC Voltage Input Module Specifications

Part Number	Description	Input Voltage	Number of Points	Response Times (ms) On	Response Times (ms) Off	Input Current	Trigger Voltage	Points per Common	Connector Type
IC693MDL241	24VAC/VDC	0-30	16	12	28	7 mA	11.5-30	16	TB
IC693MDL632	125V Pos/Neg	0-150	8	7	7	4.5 mA	90-150	4	TB
IC693MDL634	24V Pos/Neg	0-30	8	7	7	7 mA	11.5-30	8	TB
IC693MDL645	24V Pos/Neg	0-30	16	7	7	7 mA	11.5-30	16	TB
IC693MDL646	24V Pos/Neg Fast	0-30	16	1	1	7 mA	11.5-30	16	TB
IC693MDL654	5/12V Pos/Neg Fast	0-15	32	1	1	3.0 mA @5v 8.5 mA @12v	4.2-15	8	FCN**
IC693MDL655	24V Pos/Neg Fast	0-30	32	2	2	7 mA	11.5-30	8	FCN
IC693MDR390	DC In/Relay Out	-30 - +30	8	1	1	7.5 mA	15-32	8	TB
IC693ACC300	Input Simulator	N/A	8/16	20	30	N/A	N/A	16	Switches

** Terminal Block (20 Screws)*
*** Fujitsu Connector (32-pt modules have two 24-pin connectors with Connector Kits available)*

AC Voltage Output Module Specifications

Part Number	Description	Load Voltage	Number of Points	Response Time (ms) On	Response Time (ms) Off	Load Current per Point	Output Type	Points per Common	Connector Type
IC693MDL310	120V (fused)	85-132VAC	12	1	1/2cy	0.5A	Triac	6	TB*
IC693MDL350	120/240V (fused)	85-264VAC	8	1	1/2cy	2A	Triac	4	TB
IC693MDL340	120V (fused)	85-132VAC	16	1	1/2cy	0.5A	Triac	4	TB
IC693MDL390	120/240V Isolated	85-264VAC	5	1	1/2cy	2A	Triac	1	TB
IC693MDL930	120/240V Iso. N.O.	5-250VAC	8	15	15	4A	Relay	1	TB
IC693MDL931	120/240V Iso .NC/N.O.	5-250VAC	8	15	15	8A	Relay	1	TB
IC693MDL940	120/240V N.O.	5-250VAC	16	15	15	2A	Relay	4	TB
IC693MDR390	DC In/Relay Out N.O.	5-250VAC	8	15	15	2A	Relay	4	TB
IC693MAR590	AC In/Relay Out N.O.	5-250VAC	8	15	15	2A	Relay	4	TB

DC Voltage Output Module Specifications

Part Number	Description	Load Voltage	Number of Points	Response Time (ms) On	Response Time (ms) Off	Load Current per Point	Output Type	Points per Common	Connector Type
IC693MDL730	12/24V Positive (fused)	12-24VDC	8	2	2	2A	Transistor	8	TB
IC693MDL732	12/24V Positive	12-24VDC	8	2	2	0.5A	Transistor	8	TB
IC693MDL740	12/24V Positive	12-24VDC	16	2	2	0.5A	Transistor	8	TB
IC693MDL742	12/24V Positive ESCP	12-24VDC	16	2	2	1A	Transistor	8	TB
IC693MDL731	12/24V Negative (fused)	12-24VDC	8	2	2	2A	Transistor	8	TB
IC693MDL733	12/24V Negative	12-24VDC	8	2	2	0.5A	Transistor	8	TB
IC693MDL741	12/24V Negative	12-24VDC	16	2	2	0.5A	Transistor	8	TB
IC693MDL734	125V Positive/Negative	11-150VDC	6	7	5	1A	Transistor	1	TB
IC693MDL930	24V N.O. Isolated	5-30VDC	8	15	15	4A	Relay	1	TB
IC693MDL931	24V N.O./NC Isolated	5-30VDC	8	15	15	8A	Relay	1	TB
IC693MDL940	24V N.O.	5-30VDC	16	15	15	2A	Relay	4	TB
IC693MDL752	5/24V Negative	5,12-24VDC	32	.5	.5	0.5A	Transistor	8	FCN**
IC693MDL753	12/24V Positive	12-24VDC	32	.5	.5	0.5A	Transistor	8	FCN
IC693MDR390	DC In/Relay Out N.O.	5-30VDC	8	15	15	2A	Relay	4	TB
IC693MAR590	AC In/Relay Out N.O.	5-30VDC	8	15	15	2A	Relay	4	TB

* *Terminal Block (20 Screws)*
** *Fujitsu Connector (32-pt modules have two 24-pin connectors with Connector Kits available)*

Figura 19 – Especificação de Módulos de Entrada e Saída Discreta da GE-FANUC.

7.2 ENTRADAS E SAÍDAS DE DADOS NUMÉRICOS

Em geral, a interface para dados numéricos pode ser classificada em dois grupos: interfaces para dispositivos digitais multi-bit e interfaces para dispositivos analógicos.

A interface multi-bit permite que um grupo de bits possa ser tratado como uma única unidade de entrada ou saída, por exemplo, entradas/saídas BCD. A interface analógica permite que grandezas analógicas possam ser lidas pelo controlador ou que o controlador possa modificar uma grandeza analógica atuando em dispositivos especiais. Devido à evolução das interfaces homem-máquina, as chaves multi-bits estão sendo menos utilizadas.

A entrada para registro ou interface de entrada BCD fornece uma comunicação paralela entre o processador e dispositivos de entrada numéricos. Esta interface é geralmente utilizada para entrada de parâmetros em localizações específicas na memória chamada de registros. Os parâmetros de entrada típicos são valores presets de temporizadores, contadores e valores set-points.

Tabela 3 - Dispositivos de entradas/saídas numéricas

Entradas	Saídas
Transdutor de temperatura	Válvulas proporcionais
Transdutor de pressão	Atuadores
Células de carga	Registradores
Transdutores de umidade	Drivers de motores
Transdutores de fluxo	Medidores analógicos
Chaves thumbwheel	Displays 7 segmentos
Leitoras de códigos de barra	Painéis inteligentes

Esta interface geralmente aceita tensões na faixa de 5 VDC (TTL) a 24 VDC e são agrupados em um módulo contendo 16 ou 32 entradas que corresponde a 1 ou 2 registros I/O. Instruções de manipulação de dados, como GET ou similar, são utilizadas para acessar os dados numéricos de entrada.

A interface de saída numérica fornece uma comunicação paralela entre o processador de dispositivos de saída, tal como display de 7 segmentos. Instruções como PUT ou similares são utilizadas para colocar os dados disponíveis nas saídas.

A interface de entrada analógica contém os circuitos necessários para aceitar sinais analógicos de tensão ou corrente dos dispositivos de campo. A tensão ou corrente de entrada é convertida para um código digital proporcional ao valor analógico, através de um conversor analógico digital (ADC). O código digital gerado é armazenado na memória do controlador como um registro para uso posterior. A Figura 20 apresenta o diagrama de blocos de uma entrada analógica.

O valor analógico é geralmente expresso como um valor BCD ou decimal em uma faixa que dependerá da implementação realizada pelo fabricante. Por exemplo poderemos ter os seguintes casos:

- Valor analógico de 4 a 20 mA na faixa de 0 a 32000;
- Valor analógico de 0 a 5 V na faixa de 0000 a 0255.

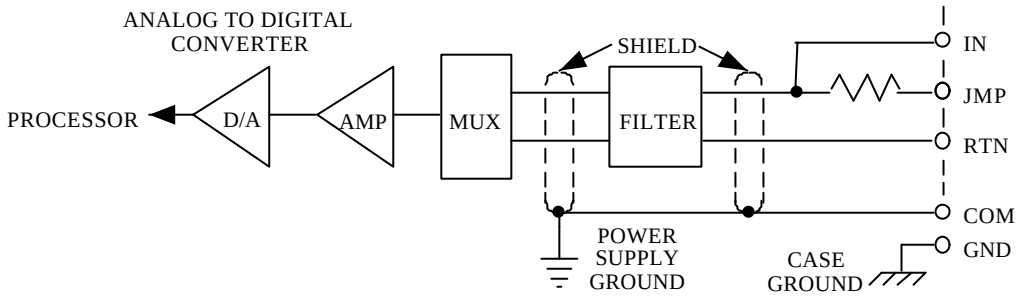


Figura 20 - Diagrama de blocos de uma entrada analógica

A interface para saídas analógicas recebe do processador dados numéricos que são convertidos em valores proporcionais de corrente ou tensão e aplicados nos dispositivos de campo. A interface contém um conversor digital analógico (DAC) e realiza a isolação através de opto-acopladores.

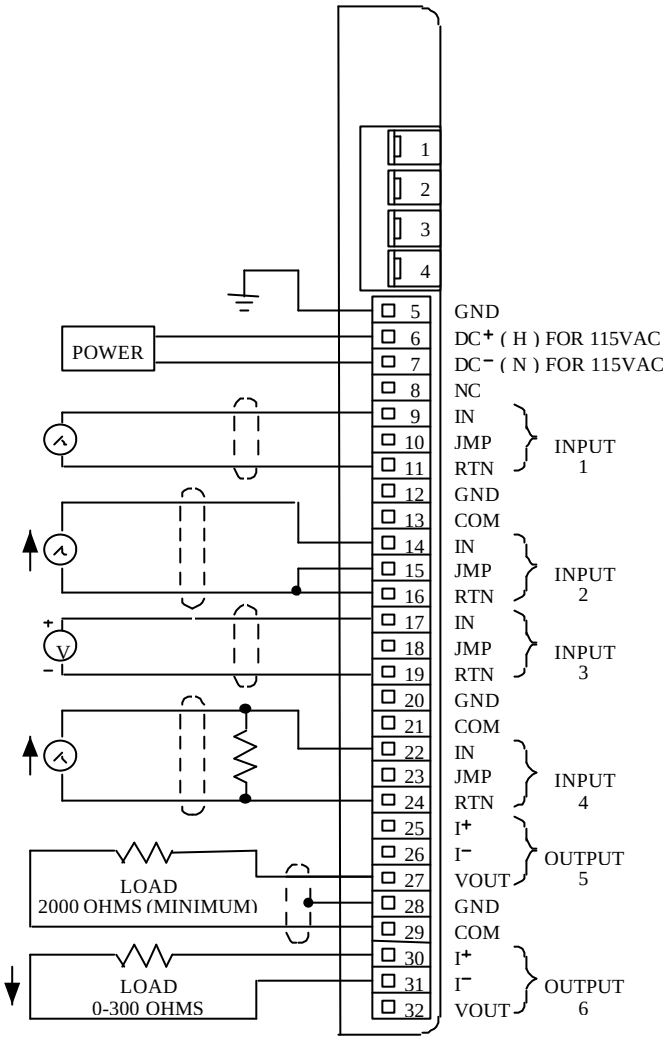


Figura 21 - Diagrama de conexão de entradas e saídas analógicas

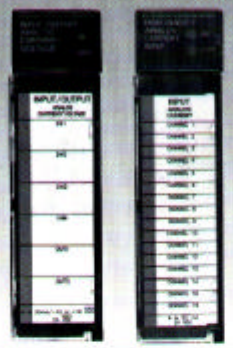
As faixas de valores de tensão e corrente para entradas e saídas analógicas mais utilizadas na indústria são: -10 a +10 VDC, -5 a +5 VDC, 0 a +5 VDC, 0 a +10 VDC, +1 a +5 VDC, 0 a 20 mA e 4 a 20 mA. Os conversores A/D e D/A normalmente são de 8, 10 ou 12 bits. As conexões de um módulo de entradas e saídas analógicas podem ser vistas na Figura 21.

A Figura 22 ilustra as características de diversos módulos de entradas e saídas analógicas da GE-FANUC.

Analog I/O Modules

A Growing Selection of Great Analog Offerings

GE Fanuc offers easy-to-use analog modules for control processes such as flow, temperature, and pressure.



Module Features

- Provides 12- or 16-bit resolution and fast direct access by PLC
- I/O updated automatically every sweep.
- All modules are software configured.
- High-density input, output, and combo modules provide advanced features, configurable mode selection per channel, and alarming.
- Alarm bits are sent to the CPU from the module for indication every sweep.

Analog Input Module Specifications

Part Number	IC693ALG220	IC693ALG221	IC693ALG222	IC693ALG223
Range	-10v to +10v all channels	4-20mA 0-20mA all channels	-10v to +10v 0 to +10v all channels	0-20mA, 4-20mA, 4-20mA Enhanced channel selectable
Number of Channels	4	4	16 single ended 8 differential	16 selectable
Update Rate	4ms all channels	2ms all channels	13ms all channels	13ms all channels
Resolution	12 bit 5mV/20μA/bit	12 bit 0-20mA, 5μA/bit 4-20mA, 4μA/bit	12 bit ± 10V, 5mV/20μA/bit 0-10V, 5mV/20μA/bit	12bit 0-20mA, 5μA/bit 4-20mA, 4μA/bit 4-20mA Enhanced, 5μA/bit
Accuracy	±10mV/40uA at 25°C (77°F)	0.1% full scale	0.25% at 25°C (77°F)	0.25% at 25°C (77°F)
Isolation	1500 volts rms field to logic side	1500 volts rms field to logic side	1500 volts rms field to logic side	1500 volts rms field to logic side
Input Impedance	>9 Megohms	250 ohms	250 ohms	250 ohms
Input Filter Response	17Hz	325Hz	200Hz	200Hz
Internal Power Used	27mA - 5V 98mA - 24V Isolated	25mA - 5V 100mA - 24V Isolated	112mA - 5V 4150mA - User Supplied 24VDC	120mA - 5V 65mA - User Supplied 24VDC

A relação de conversão entre o valor analógico e o valor digital é dada pela seguinte expressão genérica:

$$\frac{V_A - V_{AMIN}}{V_{AMAX} - V_{AMIN}} = \frac{V_D - V_{DMIN}}{V_{DMAX} - V_{DMIN}}$$

Onde:

V_A = Valor analógico correspondente ao valor digital VD

V_D = Valor digital correspondente ao valor analógico VA

V_{AMIN} = valor mínimo do sinal analógico

V_{AMAX} = valor máximo do sinal analógico

V_{DMIN} = valor mínimo do sinal digital

V_{DMAX} = valor máximo do sinal digital

Analog Output Specifications

Part Number	IC693ALG390	IC693ALG391	IC693ALG392
Range	-10v to +10V 4-20mA all channels	1-5V and 0-5V 0-20mA, 4-20mA all channels	0V to +10V, -10V to +10V 0-20mA, 4-20mA all channels selectable
Number of Channels	2	2	8
Update Rate	5ms all channels	5ms all channels	8ms all channels
Resolution	12 bit 2.5mV/bit	12 bit 0-20mA, 5μA/bit 4-20mA, 4μA/bit	16 bit 0.312mV/bit 4-20mA 0.5μA/bit 0-20mA 0.625μA/bit
Accuracy	±5mV at 25°C (77°F)	0-20mA, ±8μA at 25°C (77°F) 0-20mA, 4-20mA ±0.1% at 25°C (77°F)	0-20mA, 4-20mA ±0.1% at 25°C (77°F) 0-10V, -10F +10V ± 0.25 at 25°C (77°F)
Maximum Output Load	5mA (2K ohms)	5mA (2K ohms)	5mA(2K ohms)
Output Load Capacitance	2000pF	2000pF, Inductance 1H	2000pF, Inductance 1H
Internal Power Used	32mA - 5V 120mA - 24V Isolated	30mA - 5V 215mA - 24V Isolated.	110mA - 5V 345mA - User Supplied 24VDC
Isolation	1500 volts rms field to logic side	1500 volts rms field to logic side	1500 volts rms field to logic side

Analog Mixed Specifications

Part Number	IC693ALG442*	IC693ALG442*
	Input Specifications	Output Specifications
Range	0V to +10V, -10V to +10V 0-20mA, 4-20mA channel selectable	0V to +10V, -10V to +10V 0-20mA, 4-20mA channel selectable
Number of Channels	4 selectable	2 selectable
Update Rate	8ms all channels	4ms all channels
Resolution	12 bit 0V to +10V, 2.5mV/bit -10V to +10V, 5mV/bit 0-20mA, 4-20mA 5uA/bit	16 bit 0.312mV/bit 4-20mA 0.5μA/bit 0-20mA 0.625μA/bit
Accuracy	0.25% at 25°C (77°F)	0-20mA, 4-20mA ±0.1% at 25°C (77°F)
Isolation	1500 volts rms field to logic side	1500 volts rms field to logic side
Input Impedance	current mode - 250 ohms voltage mode - 800K ohms	
Input Filter Response	29Hz	
Maximum Output Load		5mA(2K ohms) 850 ohms
Output Load Capacitance		2000pF, Inductance 1H

IC693ALG442 is a combination input/output module

Figura 22 – Especificação de Módulos de Entradas e Saídas Analógicas da GE-FANUC.

7.3 MÓDULOS ESPECIAIS

Os módulos vistos anteriormente são os mais encontrados nas aplicações de controladores programáveis. Entretanto em algumas aplicações são necessários módulos especiais, tais como, interface para termopares, geração de mensagens, execução de algoritmos PID, comunicação em rede, etc.

Estes módulos especiais, também chamados de módulos inteligentes, incorporam um microprocessador de forma que a tarefa a ser realizada pelo módulo fica independente da varredura do processador.

7.3.1 Módulo para termopar

O módulo de entrada para termopar aceita sinais provenientes diretamente do transdutor. Um exemplo é o módulo que aceita sinais de termopares tipo J (ver Figura 23) fornecendo a compensação de junta fria internamente. A operação desta interface é similar a entrada analógica com exceção de que os sinais de baixo nível dos termopares são aceitáveis (aproximadamente 43 mV na temperatura máxima). Estes sinais são filtrados, amplificados e digitalizados por um conversor e então enviados ao processador sob o comando do programa de controle do usuário.

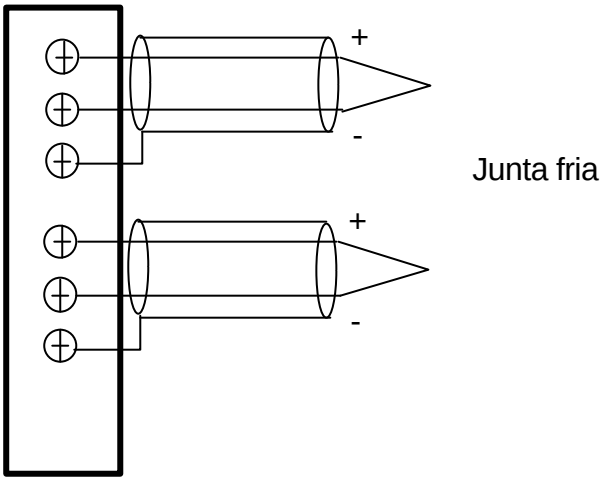


Figura 23 – Módulo de Entrada para Termopar

7.3.2 Módulo PID

Numa planta industrial, o módulo PID permite combinar o controle discreto com o controle analógico. Em algumas versões de controladores não existe um módulo específico para o controle PID (Proporcional-Integral-Derivativo) sendo que o controle é realizado pelo próprio processador, através de instruções no programa de usuário.

Quando se trata de um módulo PID, este é conectado no rack do controlador como se fosse um cartão comum. A finalidade em se empregar este módulo em uma planta industrial é controlar uma

determinada grandeza física (temperatura, vazão, etc.) continuamente de modo que essa grandeza assuma um valor preestabelecido (set-point).

O módulo é dotado de um microprocessador, memórias, conversores A/D e D/A e todo o circuito necessário para interface com os dispositivos de campo utilizados para controle. Basicamente o controle PID é um controle em malha fechada que compara o valor da variável de medida (VM) com o valor desejado ou valor de referência, também chamado de ponto de ajuste (PA). O erro resultante (e = PA - VM) é então processado seguindo uma combinação das parcelas proporcional, integral e derivativa. O resultado é um valor, variável de atuação (VA), a ser aplicado no elemento atuador. O controle atua de forma a minimizar o erro, isto é, a variável de processo tende para o valor de referência.

O módulo executa o controle independente do controlador. O controlador programável interage com o módulo através de instruções de transferências de blocos de palavras. O controlador envia ao módulo os parâmetros de ajuste de controle, tais como, ganhos, valores de set-point, valores de alarme de alta e baixa e limites máximos de saída. O controlador pode obter do módulo os dados referentes ao processo, tais como, valores de entradas e saídas analógicas, limites de alarme e diagnósticos do próprio módulo.

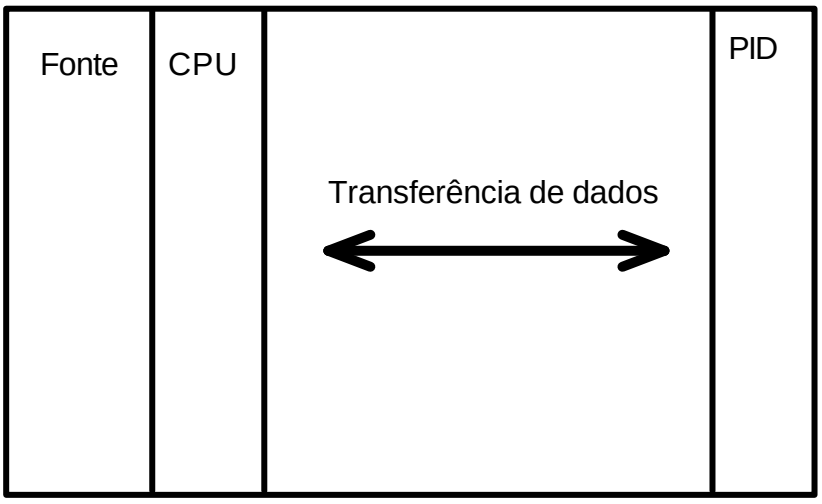


Figura 24 – Módulo PID

7.3.3 Módulos de entradas/saídas remotos

Em sistemas de maior porte é usual a instalação de módulos de entradas/saídas distante do controlador programável. Um subsistema de entradas/saídas remoto é composto por fontes de alimentação, módulos I/O e adaptadores de comunicação.

Existem duas formas de conexão dos racks remotos ao processador: configuração em barramento ou em estrela. A Figura 25 ilustra a configuração em barramento. A distância que o rack remoto pode ser colocado em relação ao processador depende do fabricante e da configuração.

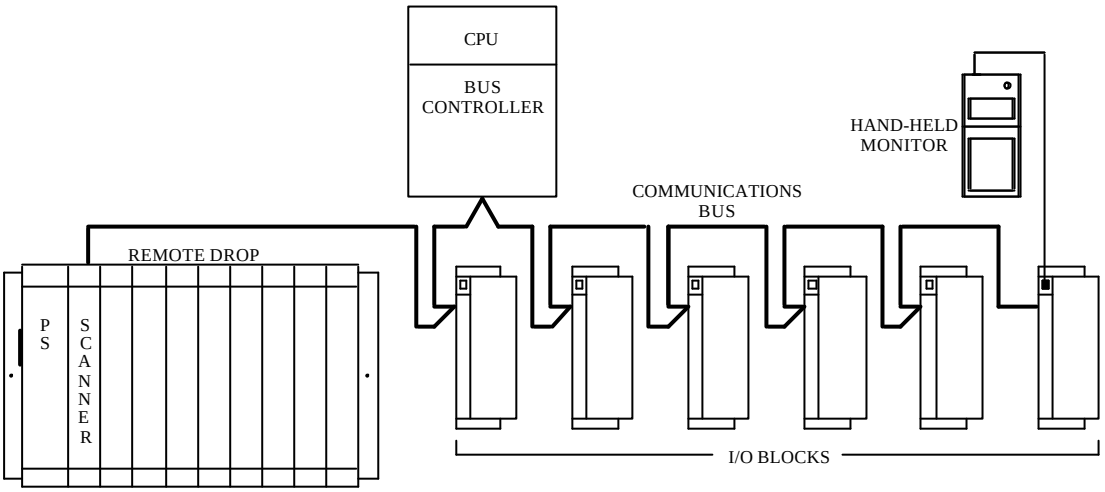


Figura 25 - Sistema de E/S remoto

As entradas/saídas remotas oferecem vantagens em termos de fiação de campo e custos de manutenção em grandes sistemas. O controlador pode ficar instalado em uma sala de controle ou em um ponto estratégico da planta conectado aos racks remotos através de um link de comunicação (cabo coaxial ou par de fios trançados). A distribuição de entradas/saídas ainda permite a instalação e o start-up de uma forma modular e independente, facilitando o entendimento e a solução dos problemas iniciais bem como as manutenções futuras.

8 LINGUAGEM DE PROGRAMAÇÃO BÁSICA

Existem 5 tipos básicos de linguagem que normalmente são encontradas em controladores programáveis e são padronizadas pela norma IEC 61131-3:

- Linguagens de relés ou diagrama de contatos;
- Linguagens por blocos funcionais;
- SFC - Sequential Function Chart (diagramas de funções seqüenciais);
- Lista de instruções;
- Texto estruturado.

Dos 5 tipos apresentados acima, a mais difundida e encontrada em quase todos os controladores é a linguagem de relés. Os blocos funcionais também podem ser encontrados com facilidade. Sendo esta última uma extensão da primeira no sentido de incluir instruções mais poderosas e tornar mais claro sua programação, esta tem sido a linguagem preferida dos fabricantes. Atualmente o SFC vem recebendo várias implementações nos melhores CP´s, afirmando-se como uma linguagem ideal para processos seqüenciais.

A linguagem de relés é uma representação gráfica da linguagem de programação do controlador. Esta linguagem é utilizada para implementar os programas de controle de máquinas e processos. Conforme o próprio nome sugere as instruções básicas se originaram no diagrama eletromecânico, cujo elemento principal de controle é o relé, especificamente sua bobina e seus contatos. Por ser a primeira linguagem utilizada pelos fabricantes é muito difundida e recebeu vários nomes desde sua criação, entre eles citamos: diagrama de escada ("ladder"), diagrama de contatos e linguagem de contatos.

Na programação de contatos de relés, dentro do formato de programação, qualquer arranjo de contatos normalmente aberto ou fechado pode ser utilizado. Pode-se fazer tantas referências quanto se desejar a uma determinada bobina através de uma instrução NA ou NF.

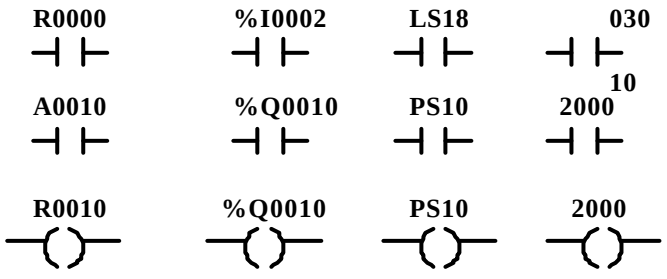
As conexões ou espaços horizontais e verticais podem ser programadas. Estas características dependem exclusivamente do ambiente utilizado para programar o controlador, isto é, do software de programação que normalmente é fornecido pelo fabricante do controlador. Observa-se porém que isto significa ocupação de área de memória. Portanto recomenda-se manter um estilo de programação que procure otimizar esta área. Procure programar instruções próximas, evitando conexões desnecessárias.

Bobinas e contatos são símbolos utilizados nesta linguagem. Os símbolos de contatos programados em uma linha representam as condições que serão avaliadas de acordo com a lógica, e como resultado determinarão o controle de saída que será representado por um símbolo de bobina. Recomenda-se que as bobinas de saída (externa ou interna) sejam programadas apenas uma vez no programa, com exceção das especiais, porém podem ser feitas tantas referências quantas forem necessárias a determinada bobina.

A forma de execução da lógica programada é muito dependente de como o fabricante implementou as instruções. Alguns fabricantes definem uma malha ou também chamada de lógica de programação. Nesta lógica podem ser programadas as instruções da linguagem e são definidas linhas de programação, onde o usuário poderá programar suas instruções básicas ou blocos, porém existe um número limite para o número de linhas que podem ser programadas. Normalmente este número de linhas define o número de bobinas que podem ser utilizadas e que podem funcionar em um formato quase paralelo.

8.1 ENDEREÇAMENTO

Na programação, cada contato e bobina é referenciado com um endereço que identifica o que está sendo avaliado e o que está sendo controlado. Estes endereços são as referências que permitem localizar na tabela de dados do controlador o estado de uma bobina interna, de saída ou ponto de entrada. Este endereço é exclusivo para cada tipo de controlador e não existe um padrão de endereçamento entre os fabricantes. Para se programar um controlador um primeiro passo é analisar o tipo de endereçamento utilizado por ele.



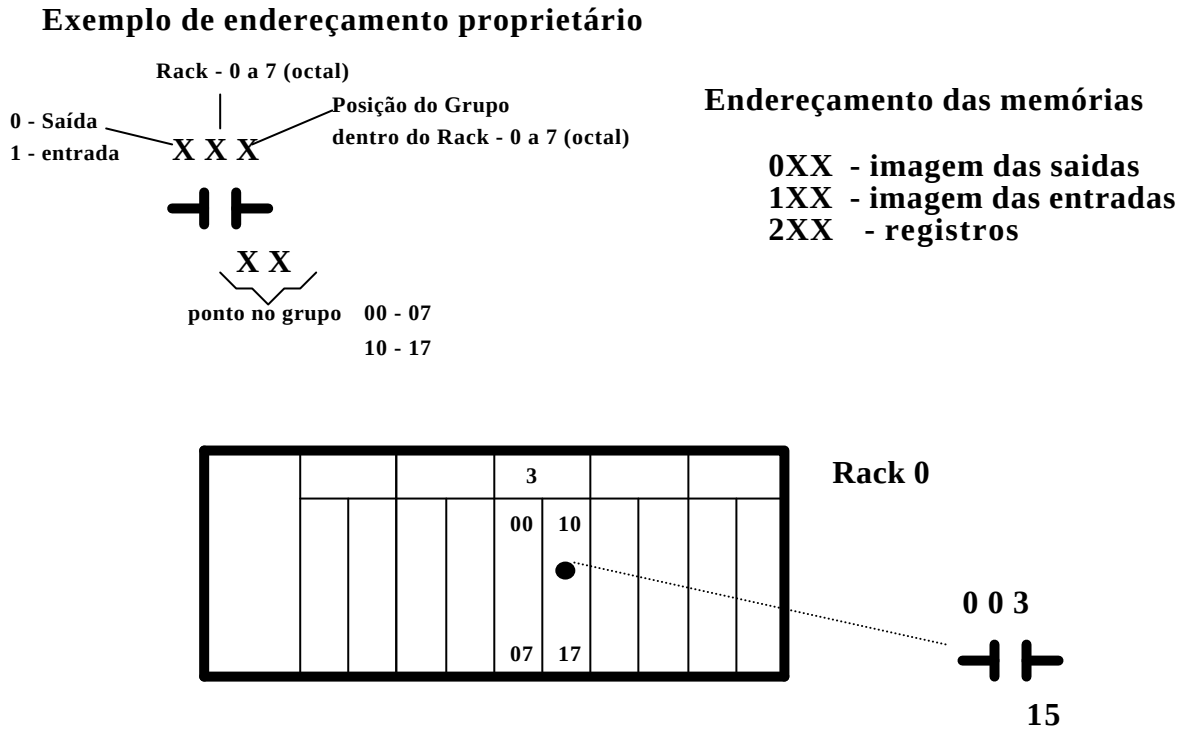


Figura 26 - Formas de endereçamento

A numeração utilizada pode ser decimal, octal ou hexadecimal. Dependendo do ambiente de programação utilizado pode-se atribuir um apelido ao endereço (*tag*, *nickname* ou etiqueta) que lembre ao programador referências com a função de "campo". A Figura 26 ilustra algumas formas de endereçamento.

A forma de endereçamento na parte inferior da figura mantém uma relação correspondente à disposição física do ponto no sistema de E/S facilitando a localização do ponto pelo pessoal de manutenção.

8.2 CONTINUIDADE LÓGICA

Para que uma saída seja energizada é necessário que exista um fluxo contínuo de energia (da esquerda para direita, de baixo para cima ou de cima para baixo) em ao menos uma linha até a bobina que representa a saída. Isto é chamado de continuidade lógica. Este conceito é similar à continuidade de corrente elétrica em um circuito eletromecânico. Embora os símbolos e instruções possam diferir muito entre os controladores, as semânticas das instruções descritas a seguir são genéricas e se aplicam a todos os controladores.

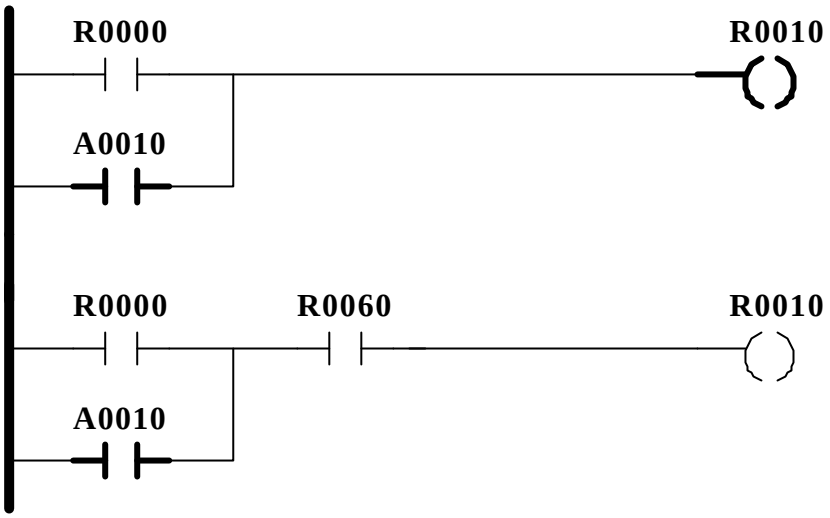


Figura 27 - Continuidade Lógica

8.3 MODELO VIRTUAL DO CLP

Para facilitar o entendimento do funcionamento do controlador programável considere que internamente existam vários relés virtuais, isto é, para cada ponto de entrada externa, saída externa e referência interna (bits internos ou relé auxiliar) existe um relé virtual conforme ilustrado na Figura 28. Cada relé virtual possui um conjunto ilimitado de contatos NA e NF.

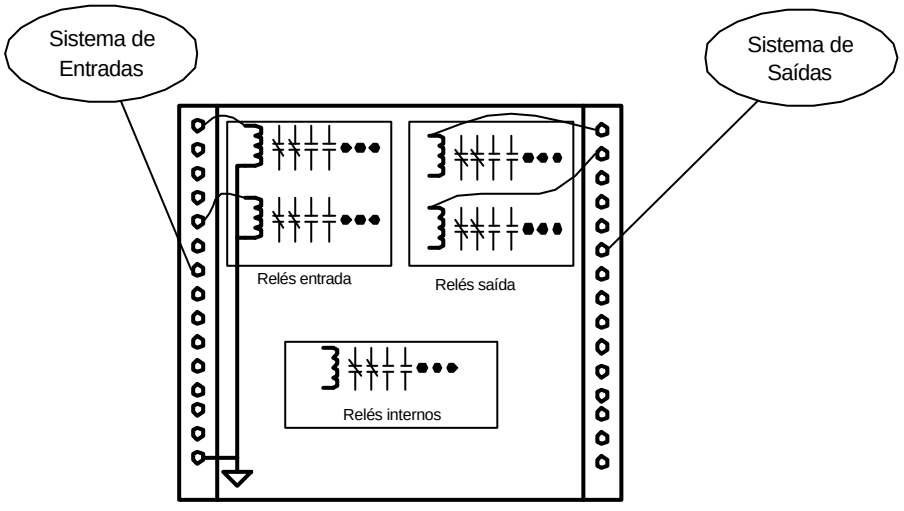


Figura 28 – Modelo Virtual do CLP

8.4 INSTRUÇÃO CONTATO NORMALMENTE ABERTO - NA

O contato normalmente aberto é programado quando a presença do sinal referenciado pelo endereço é necessário para ligar uma saída. Quando o controlador encontra esta instrução, o estado do bit cujo endereço é referenciado na instrução é examinado para uma condição verdadeira ou energizada - ON. Se o estado for verdadeiro ou energizado o contato normalmente aberto fechará e permitirá a continuidade lógica (fluxo de energia). Se o estado for falso ou desligado - OFF, então o contato normalmente aberto manterá o seu estado programado (OFF) não permitindo o fluxo de energia.

Pode-se raciocinar com o estado da BOBINA, de forma similar ao relé eletromecânico, pois quando a bobina do relé eletromecânico é energizada os contatos NA fecham, permitindo continuidade de corrente elétrica.

O endereço referenciado pode representar uma entrada externa, saída externa ou saída interna.

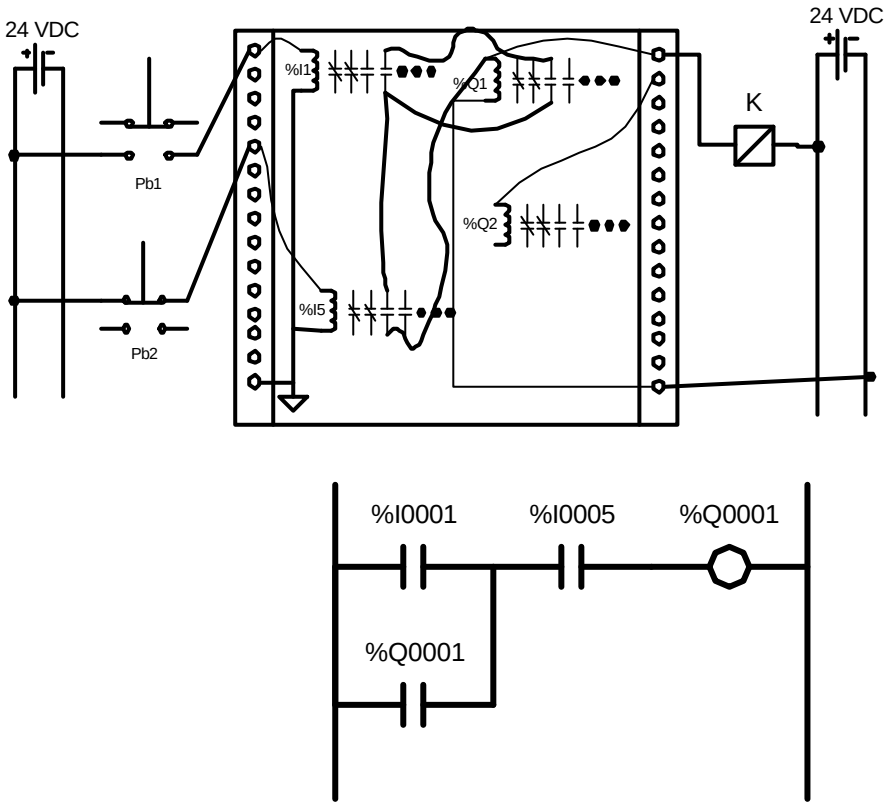


Figura 29 - Exemplo de Instrução NA

8.5 INSTRUÇÃO CONTATO NORMALMENTE FECHADO - NF

O contato normalmente fechado é programado quando a ausência do sinal referenciado é necessária para que a saída seja ligada ou energizada - ON. Quando o controlador encontra esta instrução, o endereço referenciado é examinado para uma condição de desligado ou desenergizado. Se, quando examinado, o endereço referenciado está desenergizado, então o contato normalmente fechado permanecerá fechado permitindo a continuidade lógica. Se o endereço referenciado é ON, o contato normalmente fechado abrirá interrompendo o fluxo de energia.

De forma similar podemos raciocinar com o estado da BOBINA de um relé eletromecânico, onde se a sua bobina estiver desenergizada os seus contatos NF permanecerão fechados permitindo a continuidade da corrente elétrica.

O endereço referenciado pode representar uma entrada externa, saída externa ou saída interna.

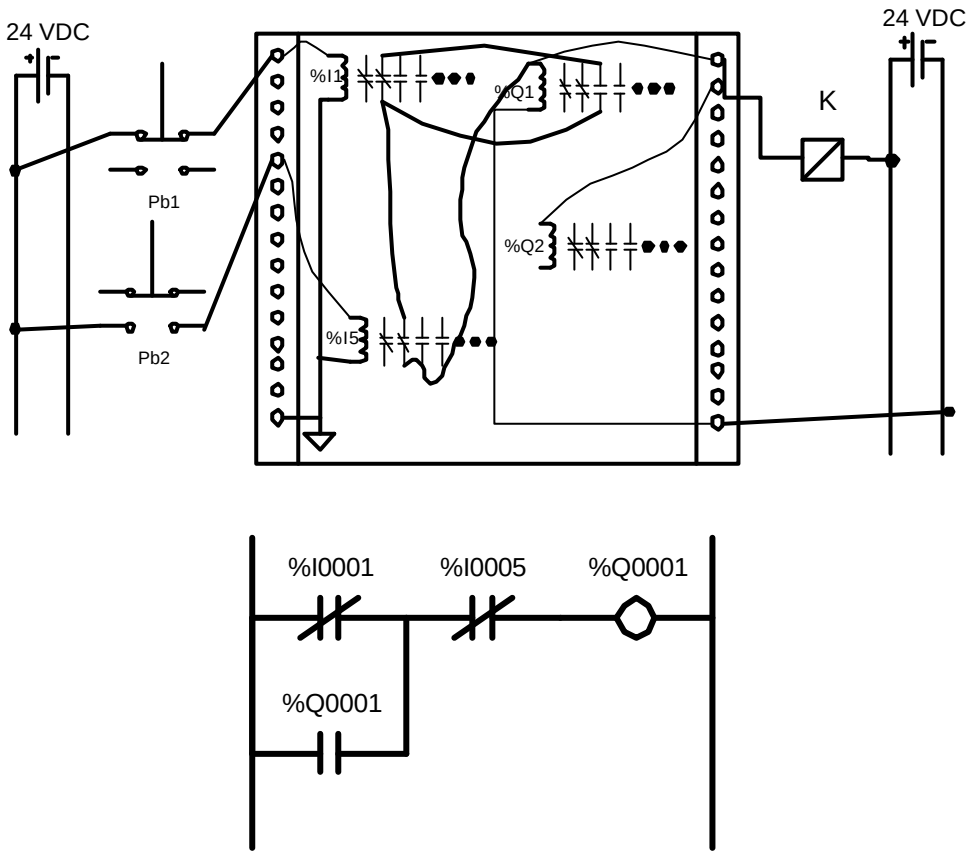


Figura 30 - Exemplo da Instrução NF

8.6 INSTRUÇÃO DE ENERGIZAR BOBINA

Esta instrução é programada para controlar tanto uma saída externa quanto uma bobina interna. Se a linha à qual está conectada tem continuidade lógica, a saída referenciada é energizada. Neste caso, os contatos NA com o mesmo endereço fecharão e os contatos NF abrirão. Se a saída é desenergizada, qualquer contato normalmente aberto abrirá e qualquer contato normalmente fechado fechará.

8.6.1 Exemplo de programação: Partida de um motor

A Figura 31 ilustra o esquema de controle de partida de um motor utilizando relés eletromecânicos. Pede-se o esquema de ligação e o programa do controlador programável para realizar a mesma função.

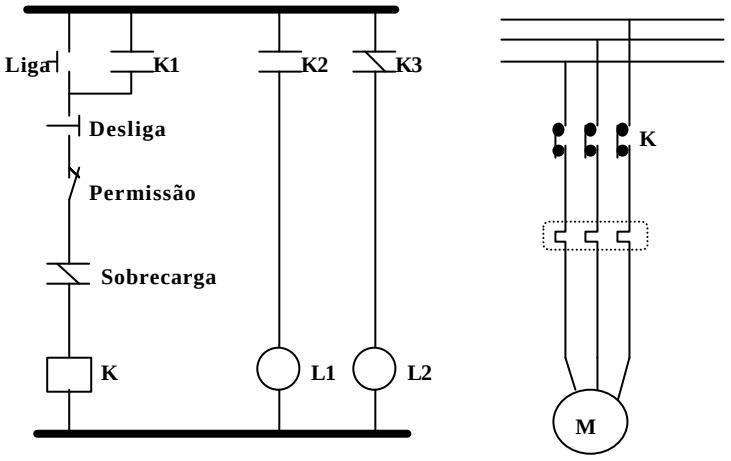


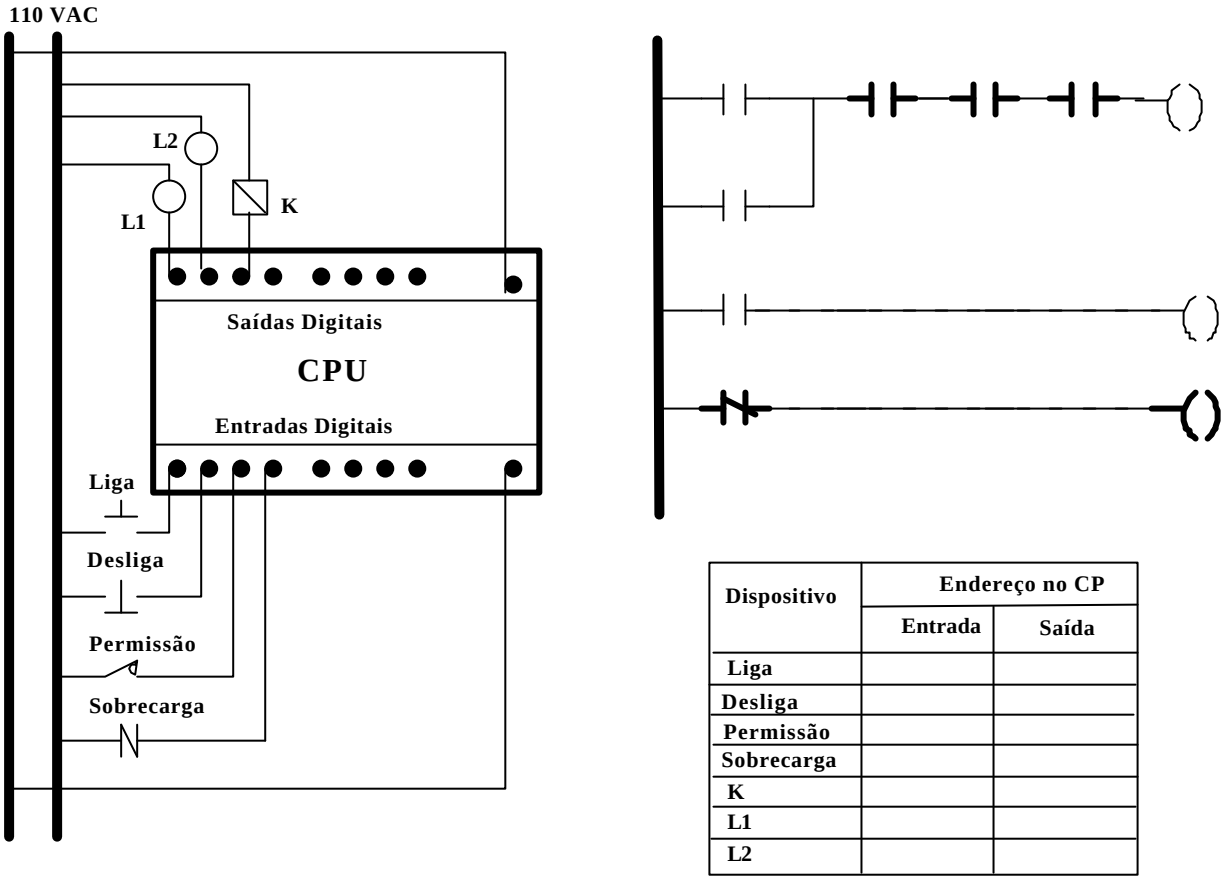
Figura 31 - Partida de Motor com circuito convencional

Solução:

Quando se considera o controlador programável como elemento de controle em uma aplicação deve-se inicialmente identificar quais serão as saídas e entradas para o controlador. As chaves e botoeiras externas deverão ser adequadamente conectadas ao módulo do CP, bem como os elementos de atuação e sinalização. No caso, teremos como entrada para o controlador as botoeiras Liga, Desliga, Permissão e o Contato de Sobrecarga.

Como saída teremos as lâmpadas de sinalização L1 e L2 e o contator do motor. Neste caso uma saída do CP é utilizada para energizar a bobina do contator de potência.

A Figura 32 ilustra o hardware e o software necessário para o controlador.



Dispositivo	Endereço no CP	
	Entrada	Saída
Liga		
Desliga		
Permissão		
Sobrecarga		
K		
L1		
L2		

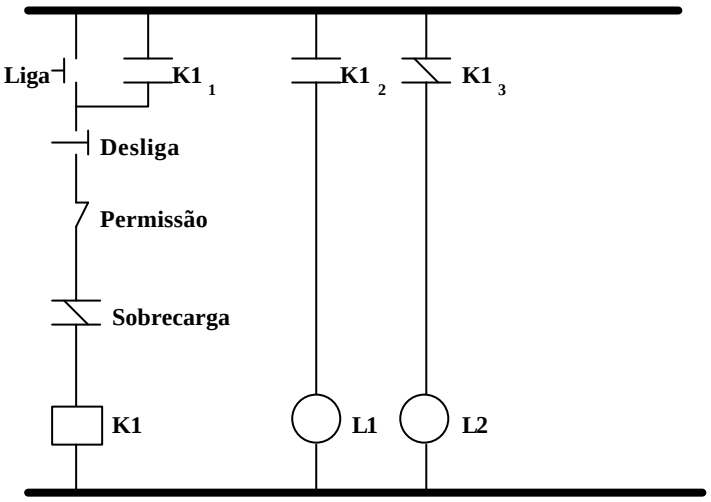
Figura 32 - Solução através de Controlador Programável

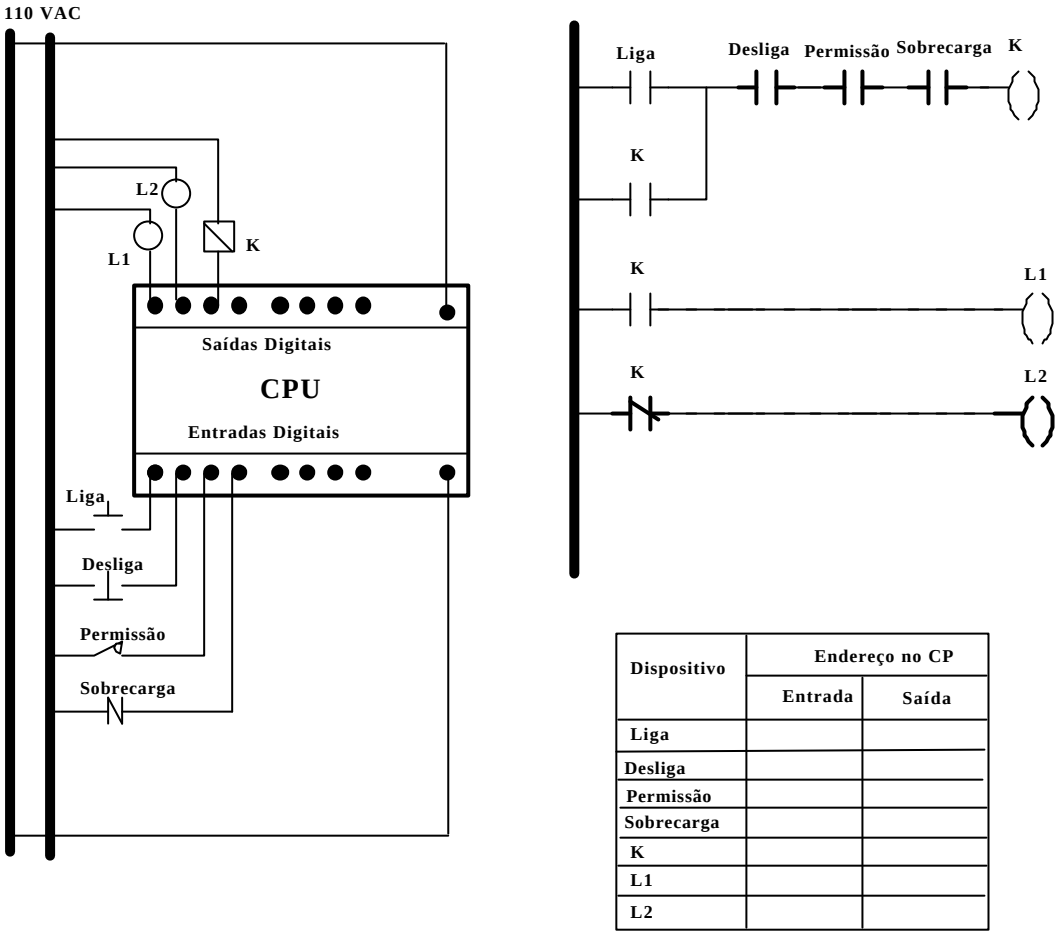
8.6.2 Alteração do exemplo anterior

Na solução anterior foi considerado que uma vez dado o comando para ligar o contator de potência este ligará. Esta solução não prevê que pode ocorrer uma falha externa na energização da bobina do contator, neste caso ocorrerá uma inconsistência de estado pois o CP operará como se o motor estivesse ligado mas por problemas fora do CP isto não será verdade. Para evitar esta situação deve ser previsto uma realimentação do estado do contator. Isto implica em utilizar um contato auxiliar do contator como um ponto de entrada no módulo do CP.

Uma terceira alteração que pode ser feita é considerar a necessidade de sinalizar ao operador o estado da *permissão*, isto é, colocar uma lâmpada para sinalizar quando se pode ligar o motor.

Realizar esta modificação e comparar com a solução convencional implementada em relés. Acrescentar no circuito anterior uma sinalização para indicar o estado da sobrecarga e comparar a solução obtida por relé e por CP.





Dispositivo	Endereço no CP	
	Entrada	Saída
Liga		
Desliga		
Permissão		
Sobrecarga		
K		
L1		
L2		

Figura 33 - Circuito modificado de partida de um motor.

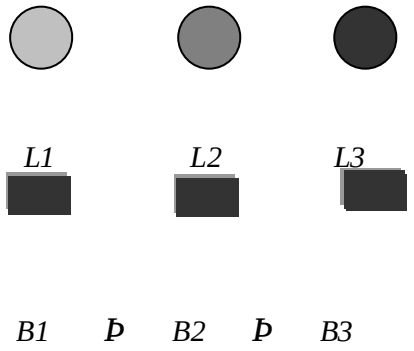
8.6.3 Proposta de Exercício

Considere que no exemplo anterior deseja-se modificar o circuito para controlar o motor através de uma chave chamada de JOG que liga o motor ao ser pressionada e desliga o motor ao ser despressionada. Também deverá ser evitado acionar o motor quando a condição de sobrecarga estiver ativada e o motor somente poderá ligar se a condição permissão estiver ativa.

Solução:

8.6.4 Exemplo de programação: Lâmpadas Seqüenciais

Considere as três lâmpadas e suas respectivas botoeiras mostradas na figura abaixo. Deseja-se implementar um circuito de controle de acionamento das lâmpadas tal que elas sejam acionadas na ordem mostrada, isto é, elas devem ser acesas e apagadas da esquerda para a direita. Uma vez iniciada a seqüência ela deve ser finalizada. Qualquer operação fora da seqüência deve ser ignorada. Considere todas as lâmpadas apagadas no estado inicial.



Solução:

8.6.5 Implementação prática

Implementar os programas desenvolvidos no ambiente de desenvolvimento Isagraf. Siga os passos fornecidos no item 9 Prática com o ISAGRAF.

8.7 INSTRUÇÃO DE ENERGIZAR BOBINA COM RETENÇÃO

Esta instrução é uma variação da anterior, incluindo uma característica similar a encontrada em flips-flops. Esta instrução deve ser programada quando for necessário manter a saída energizada após a continuidade da linha de programação ser perdida. Isto é, uma vez energizada a bobina permanecerá energizada até que um comando a desligue (ver próxima instrução).

O símbolo utilizado para esta instrução é o mesmo da bobina comum, com adição de uma letra interna ao símbolo. Alguns fabricantes utilizam "L" de liga ou “L” do inglês "*latch*" ou o "S" do inglês "set". A Figura 34 ilustra esta instrução e o gráfico de estado no tempo.

8.8 INSTRUÇÃO DE DESENERGIZAR BOBINA COM RETENÇÃO

Esta instrução é usada sempre que se quer desenergizar uma bobina com retenção, sendo a única forma de fazê-lo, quando a linha do programa apresentar continuidade lógica.

O símbolo para esta instrução é o mesmo para bobina convencional adicionando-se uma letra. Alguns fabricantes utilizam “D” de desliga ou do inglês “U” (*unlatch*) ou “R” (*reset*) respectivamente. A Figura 34 ilustra uma aplicação desta instrução e seu diagrama de tempo.

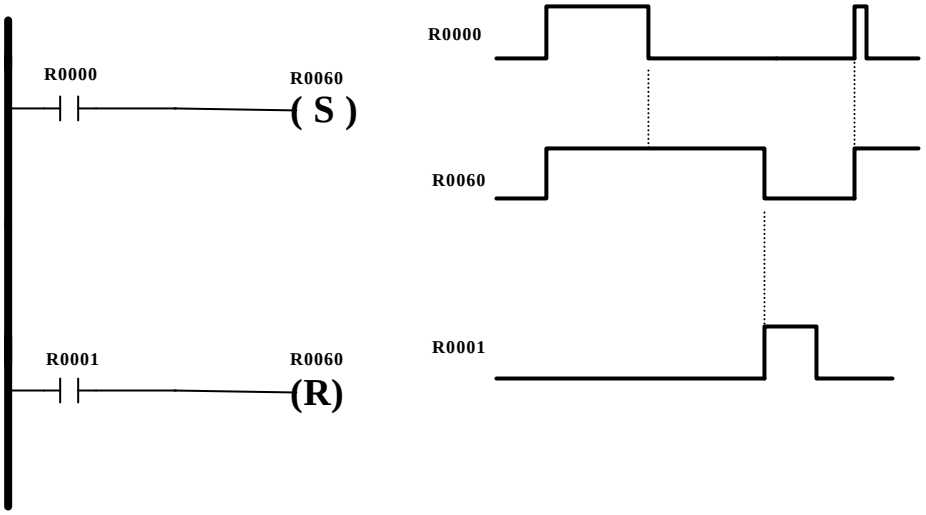


Figura 34 - Ilustração da instrução de bobina com retenção

8.9 OUTROS TIPOS DE BOBINAS

Dependendo do CLP são disponibilizados outros tipos de bobina, as mais utilizadas são:

- Bobina negada –(\ ) : complementar a bobina convencional
- Bobina retentiva – (M) : o estado da bobina é memorizado em casos de falta de energia
- Bobina pulso – (↑) : a bobina é energizada pelo período de um ciclo de varredura

8.10 CONEXÃO DE CHAVE NA E NF AO CLP

A conexão de uma chave com dois contatos, NA e NF, é implementada no CP através de uma única ligação. A lógica adequada para o funcionamento é programada pelo usuário de acordo com a sua necessidade e com o tipo de contato conectado na entrada.

A Figura 35(a) ilustra um exemplo onde a chave PB10 tem dois contatos, NA e NF. Deseja-se utilizar o contato NA para energizar a lâmpada piloto PL20, enquanto o contato NF energizará a lâmpada PL21. Na implementação deste circuito no CP, somente uma entrada precisará ser conectado à interface de entrada, mesmo que a chave tenha dois contatos. Uma vez conectado, o usuário programa o CP para executar a função requerida. É indiferente o tipo de contato conectado, sempre se poderá programar o CP através de uma única informação.

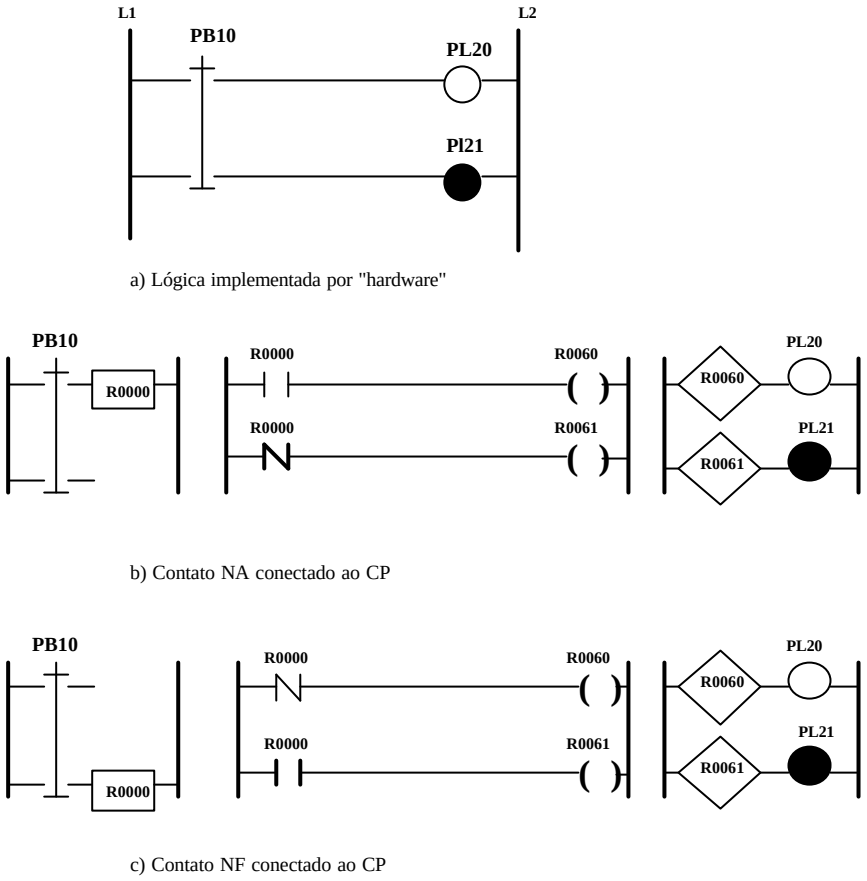


Figura 35 - Conexões típicas de chaves NA e NF no CP

Se o contato NA de PB10 é conectado no módulo de entrada, a programação é feita de acordo com a Figura 35(b). Se o contato NF é conectado ao módulo a programação é a inversa como ilustrado na Figura 35(c). A lógica de programação é invertida porque durante a operação normal (chave pressionada) o contato de PB10 está aberto e deve ser testado por um estado lógico 1 ou seja se energizado, necessitando portanto de um instrução NA. E quando se deseja testar por um estado lógico 0, se desenergizado, utiliza-se a instrução NF.

8.11 INSTRUÇÃO TEMPORIZADOR

A instrução temporizador é uma instrução que realiza a mesma função do relé de tempo e outros dispositivos temporizadores. Geralmente condicionados por instruções do tipo NA e NF os temporizadores quando habilitados geram um intervalo de tempo e ativam um bit no final do intervalo.

Cada instrução de temporização tem dois registros associados que devem armazenar o valor pré-selecionado e o valor acumulado. Portanto, um temporizador sempre irá ocupar dois registros do controlador. Estes registros são definidos da seguinte forma:

- Valor acumulado (AC): armazena o valor do tempo decorrido desde a habilitação do temporizador, isto é, a energização da bobina do temporizador.
- Valor Pré-selecionado (PR): este valor deve ser definido pelo usuário, define o intervalo de tempo desejado.

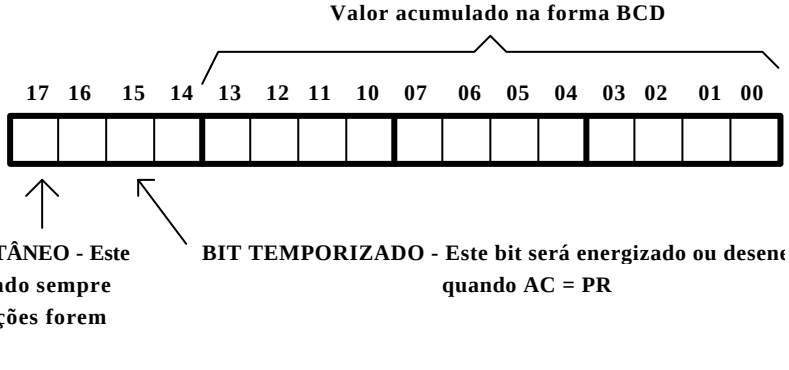
Estes valores são armazenados na tabela de dados no formato BCD (decimal codificado em binário) ou ponto flutuante. A base de tempo também pode variar de controlador para controlador, sendo que alguns permitem a seleção na instrução e outros mantêm uma base de tempo fixa. Normalmente a base de tempo é definida entre 0.01, 0.1, e 1 segundos (depende do tipo de CP).

O temporizador conta intervalos de tempo transcorridos na base de tempo selecionada e armazena essa contagem no registro definido para valor acumulado (AC). Quando o valor AC for igual ao valor PR, um bit é energizado ou desenergizado, dependendo do tipo de instrução de temporizador. Este bit, normalmente, é utilizado como contatos NA ou NF, ou bobinas externas ou internas.

A seguir são apresentados dois exemplos de instrução temporizador encontrados em controladores programáveis.

8.11.1 Exemplo de temporizador baseado em contatos de relés

Neste caso a instrução temporizador é implementada utilizando os símbolos de bobinas e contatos de relés. A Figura 36 abaixo ilustra como está organizado o registro que controla o temporizador e um exemplo de programação.



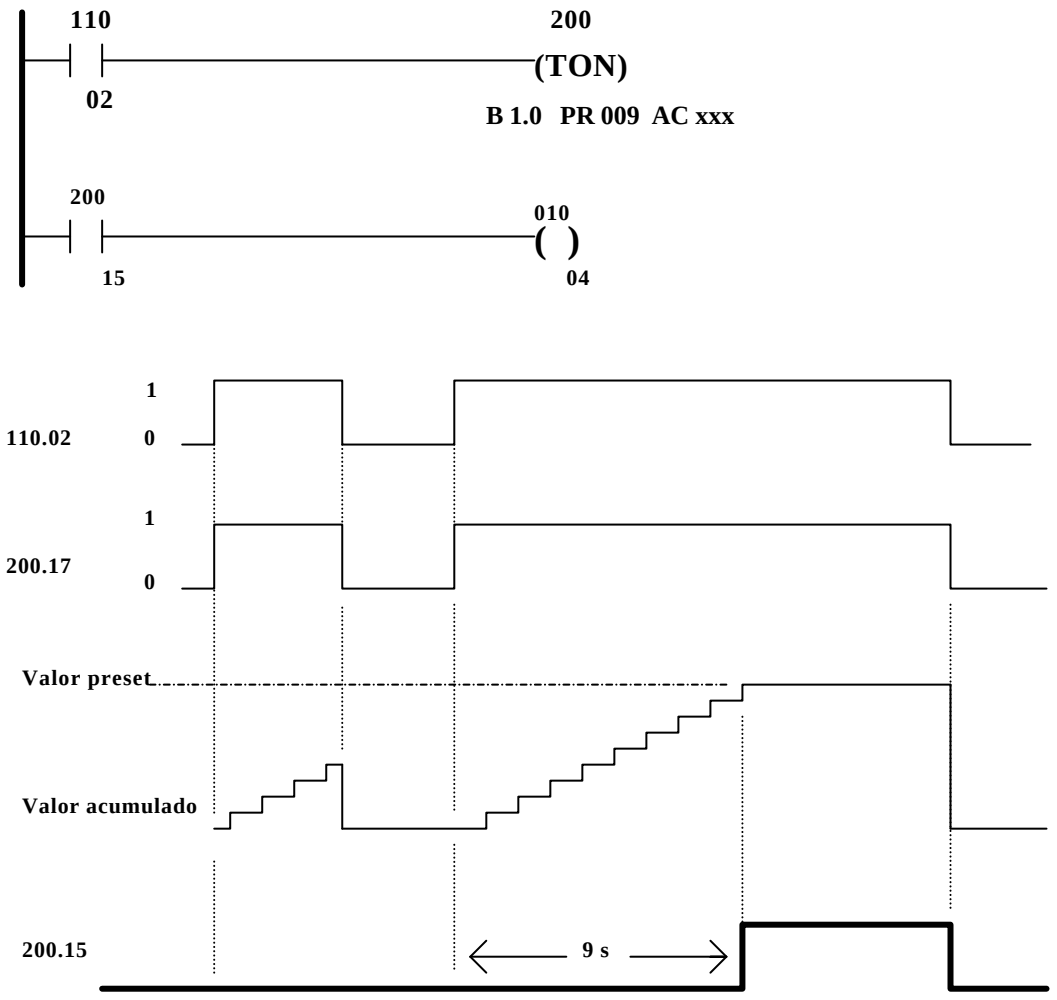
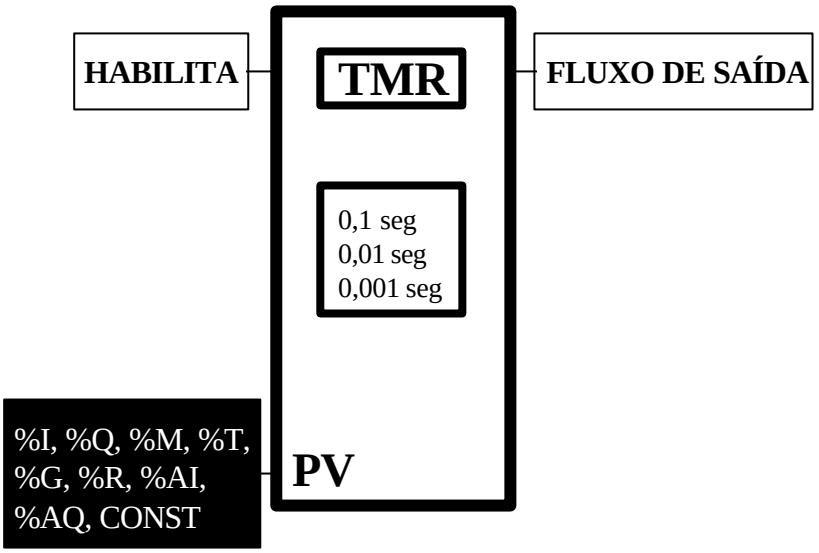


Figura 36 - Instrução de temporizador baseado em contatos de relés

8.11.2 Bloco temporizador do controlador programável GE-FANUC 9030

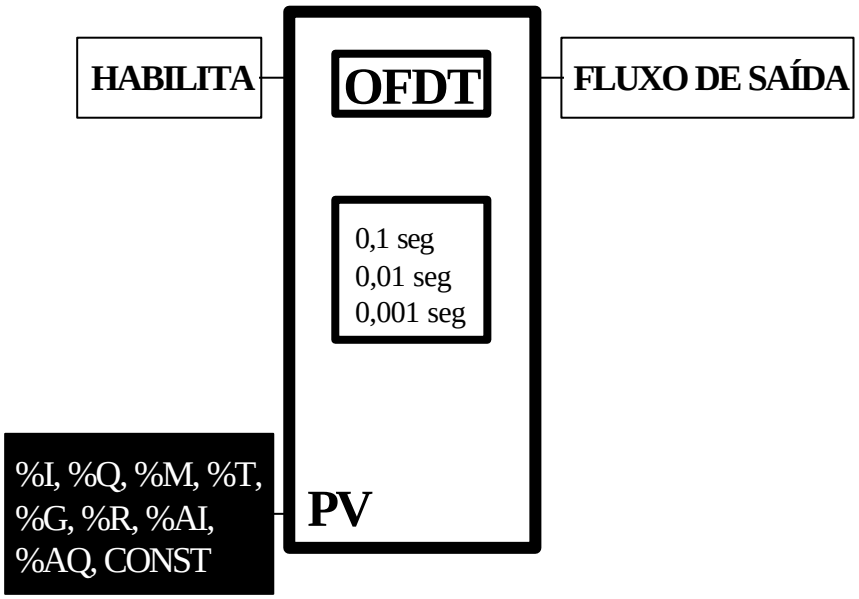
Blocos funcionais são instruções de alto nível que permitem ao usuário programar funções mais complexas usando a Linguagem de Relés. Além do que, os blocos facilitam a programação e a depuração do programa.

As instruções básicas NA e NF são utilizadas para habilitar ou não os blocos funcionais. Os blocos possuem registros para armazenamento de dados utilizados pela instrução: valor pré-set, valor acumulado, base de tempo, etc.



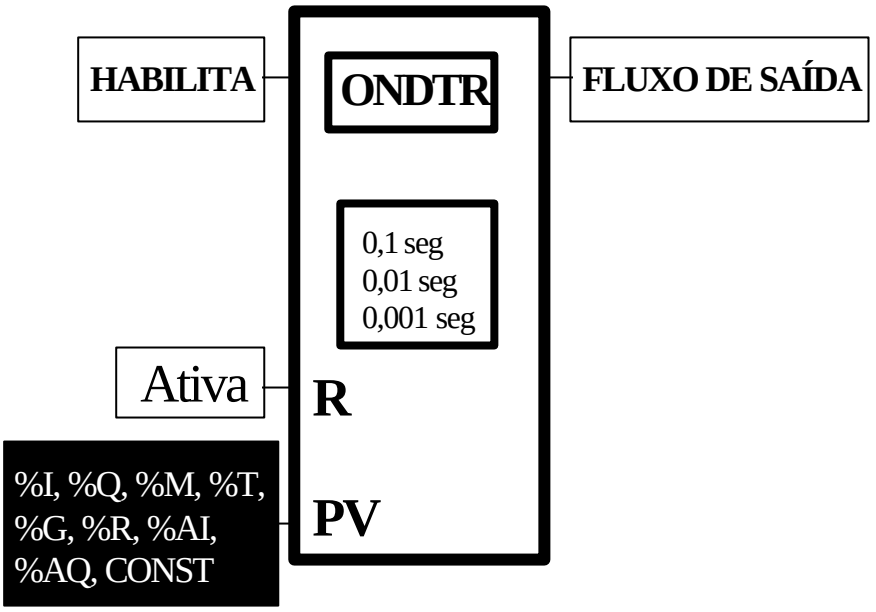
???????
%R Referência interna
CV Valor corrente armazenado em %R
PV Valor preset armazenado em %R + 1
Palavra de Controle Armazenado em %R + 2

Figura 37 – Bloco Temporizador não-retentivo do CP GE-FANUC 9030



???????
%R Referência interna
CV Valor corrente armazenado em %R
PV Valor preset armazenado em %R + 1
Palavra de Controle Armazenado em %R + 2

Figura 38 – Temporizador na desenergização do CP GE-FANUC 9030



???????
%R Referência interna
CV Valor corrente armazenado em %R
PV Valor preset armazenado em %R + 1
Palavra de Controle Armazenado em %R + 2

Figura 39 – Temporizador retentivo do CP GE-FANUC 9030

8.11.3 Exemplo de temporizador baseado em bloco funcional (IEC61131-3)

A Figura 40 ilustra um bloco genérico onde observamos posições ou células do bloco que devem ser definidas pelo programador na configuração do bloco. O número de entradas e saídas dependerá do bloco funcional, as entradas representam linhas de controle do bloco que serão ativadas ou não de acordo com a lógica que o usuário quer implementar. As saídas representam o resultado lógico da operação do bloco e são utilizadas para energizar bobinas ou entradas de outros blocos.

O funcionamento do bloco segue o conceito já introduzido anteriormente. Quando o bloco estiver no estado de energizado (entrada IN energizada) o registro que contém o valor acumulado ET é incrementado segundo a base de tempo. Quando o valor ET for igual ao valor PT, valor pré-selecionado, a saída EQ do bloco é energizada.

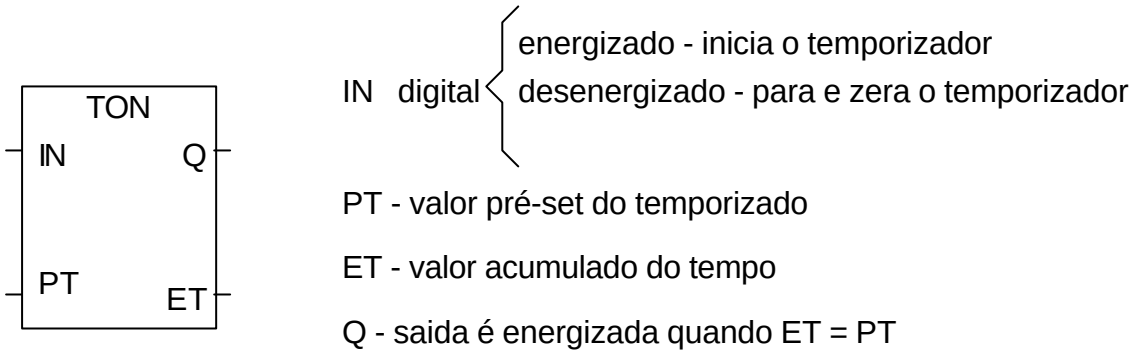


Figura 40 - Exemplo de um bloco temporizador genérico (IEC 61131-3)

A saída Q poderá fornecer "energia" a qualquer elemento conectado à direita do bloco, observando o funcionamento do temporizador. A Figura 41 ilustra exemplos de aplicação do bloco temporizador.

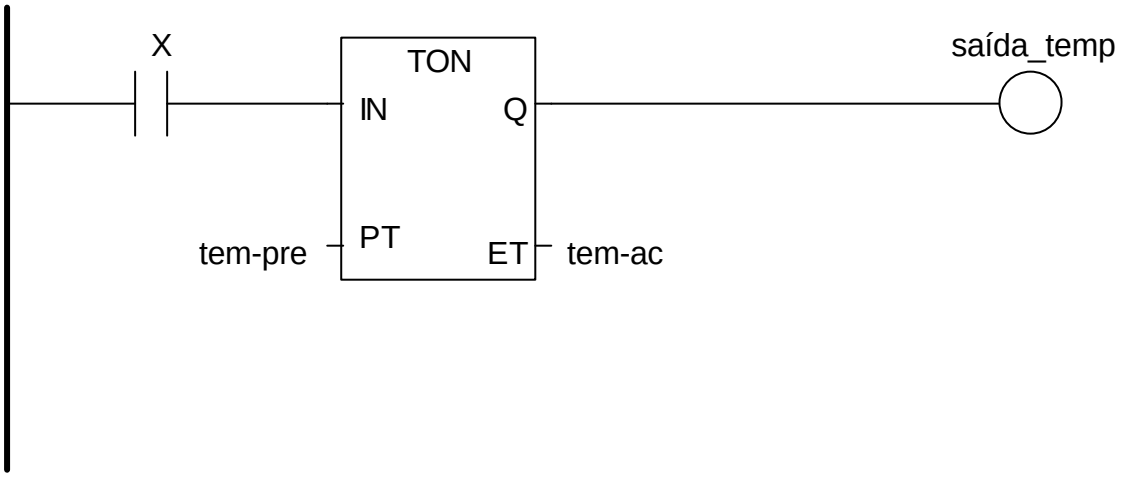


Figura 41 - Exemplo de aplicação do bloco Temporizador

8.11.4 Implementação de um desligamento temporizado no exemplo do motor

Modifique o programa desenvolvido no Isagraf de forma que o motor uma vez acionado possa ser desligado automaticamente após um certo tempo, por exemplo, 10 s.

Solução:

8.11.5 Partida estrela-triangulo de motor

Faça um programa para a partida estrela triangulo de um motor e implemente no Isagraf.

Solução:

8.12 INSTRUÇÃO CONTADOR

A instrução contador é utilizada para contar eventos que são representados no controlador como transições de falso para verdadeiro. A instrução armazena este valor no seu valor acumulado AC. Existem três tipos de instrução de contador: contador crescente, contador decrescente e contador bidirecional.

As instruções de contador diferem das instruções de temporizador pelo fato de não terem base de tempo.

A instrução contador crescente incrementa o valor acumulado AC para cada transição falso/verdadeiro da linha. Quando o acumulador atinge o valor presetado sua saída é energizada e a instrução continua a incrementar seu valor AC.

A instrução de contador decrescente subtrai uma unidade do seu valor acumulado para cada transição falso/verdadeiro da linha. Se o valor AC cair abaixo de 000, sua saída é energizada para indicar esta condição.

Em alguns controladores a instrução de contador decrescente forma um par com a instrução de contador crescente, obtendo-se assim um contador bidirecional.

8.12.1 Exemplo de instrução contador baseada em contatos de relés

Os quatros bits restantes no registro de valor acumulado - AC são utilizados como bits de estado, sendo:

- ***bit 14*** *(bit de overflow/underflow) - é energizado quando o valor AC do CTU exceder 999 ou o valor AC do CTD passar abaixo de 000.*

- ***bit 15*** *(bit de executado) - é energizado quando a contagem for alcançada ou excedida, isto é quando o valor AC for maior ou igual ao valor PR.*

- ***bit 16*** *(bit instantâneo CTD) - é energizado quando a condição da linha da instrução CTD for verdadeira.*

- ***bit 17*** *(bit instantâneo CTU) - é energizado quando a condição da linha da instrução CTU for verdadeira.*

Contador Crescente

Esta instrução incrementa o valor acumulado AC para cada transição falso/verdadeiro da linha. Quando o acumulador atinge o valor presetado o bit 15 é energizado e a instrução continua a incrementar seu valor AC. O bit 17 é energizado de acordo com a continuidade lógica da linha.

Se o valor AC passar de 999, o bit 14 é energizado para indicar uma condição de overflow e o CTU continua a contar no sentido crescente a partir de 000. o bit 14 pode ser examinado para colocar contadores em cascata para contagens maiores que 999.

Instrução de rearme do contador

A instrução de Rearme do Contador - (CTR) permite zerar o valor acumulado e levar o estado dos bits de controle para zero. Esta instrução é ativada na presença da continuidade lógica da linha, deve-se usar o mesmo endereço do registro que a instrução CTU. Quando programado os valores AC e PR aparecem automaticamente na instrução.

Instrução de Contador decrescente

A instrução de contador decrescente (CTD) subtrai uma unidade do seu valor acumulado para cada transição falso/verdadeiro da linha. Cada vez que a linha CTD passar a verdadeira, o bit 16 é energizado. Quando o valor acumulado for igual ou maior que o valor PR, o bit 15 é energizado. Se o valor AC cair abaixo de 000, o bit 14 é energizado para indicar uma condição de underflow e o CDT continua a contar no sentido decrescente a partir de 999.

Normalmente, a instrução de Contador Decrescente forma um par com a instrução de Contador Crescente, obtendo-se assim um contador bidirecional. Neste caso deve-se utilizar o mesmo endereço de registro, valor AC e PR. A Figura 42 ilustra este programa.

Figura 42 - Contador Crescente/Decrescente

8.12.2 Exemplo de instrução contador baseada em bloco funcional

O bloco Contador tem uma operação semelhante ao temporizador, porém a sua função é contar eventos, isto é, transições falsas/verdadeiras na linha de controle. O formato do bloco é o mesmo, eliminando a posição referente à base de tempo. A Figura 43 ilustra o bloco Contador. A saída superior é energizada sempre que o valor acumulado - CV for igual ao valor preset - PV.

A entrada intermediária irá controlar o rearme do contador, se esta entrada for energizada o contador é "resetado". Esta entrada é dominante, se estiver ativa o contador permanecerá resetado.

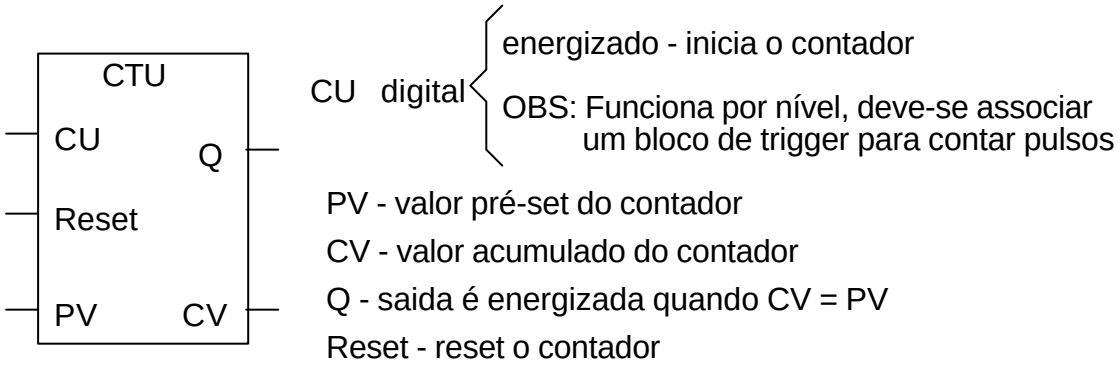


Figura 43 - Bloco contador

A entrada CU funciona por nível, isto é, a entrada não deteta a borda de subida do pulso. É preciso associar um outro bloco funcional para conseguir o efeito da contagem de eventos. A Figura 44 ilustra um exemplo do uso do bloco contador de pulsos.

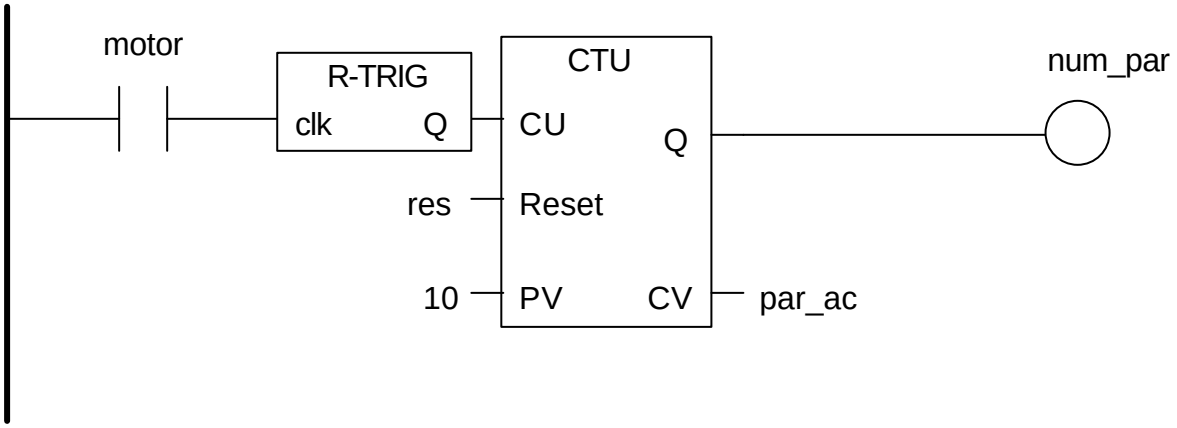
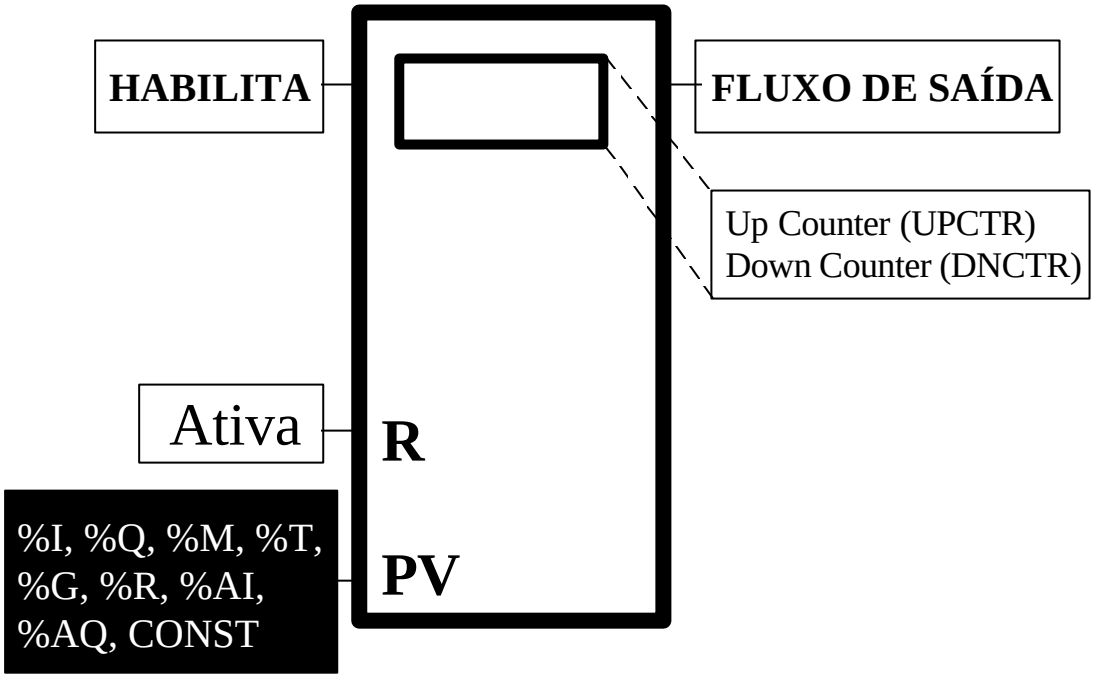


Figura 44 – Associação do bloco trigger para o contador de pulsos

8.12.3 Bloco contador do controlador programável GE FANUC 9030

A Figura 45 ilustra o bloco contador do GE-FANUC. A entrada do bloco detecta o pulso não sendo necessário qualquer bloco adicional para a contagem de eventos. O contador é retentivo, somente quando a entrada R for ativada é que o valor acumulado será resetado.



???????	
%R	*Referência interna*
CV	*Valor corrente armazenado em %R*
PV	*Valor preset armazenado em %R + 1*
Palavra de Controle	*Armazenado em %R + 2*

Figura 45 – Bloco contador do Controlador programável GE-FANUC

8.12.4 Proposta de Exercício prático

No exercício de partida do motor, colocar uma lógica de programação que impeça o motor de partir 2 vezes seguida no período de 10 s.

Solução:

8.13 OUTRAS INSTRUÇÕES

O conjunto de instruções em Linguagem Ladder pode se extender de forma a englobar vários blocos funcionais. A diversidade de funções depende do fabricante do controlador. A seguir serão descritas algumas funções do CLP GE FANUC

8.13.1 Instruções aritméticas

As instruções de manipulação de dados são instruções de capacidade mais avançada que vieram para aumentar ainda mais a versatilidade dos controladores. As instruções vistas anteriormente estavam limitadas a operações do tipo bit. As instruções de manipulação de dados permitem a operação de registros, ou seja conjuntos de bits.

As instruções aritméticas, na forma de blocos, são um exemplo desta aplicação. A Figura 46 ilustra o bloco de funções aritméticas do controlador GE FANUC. Os três operandos necessários são definidos nas três posições do bloco, sendo que as posições I1 e I2 manterão seus operandos e a posição Q armazenará o resultado da operação. O bloco é habilitado através de sua entrada e dependendo da operação e seu resultado, as saídas são ativadas. As posições 1, 2 e 3 podem ser constantes, posições de memória ou as referência indicadas.

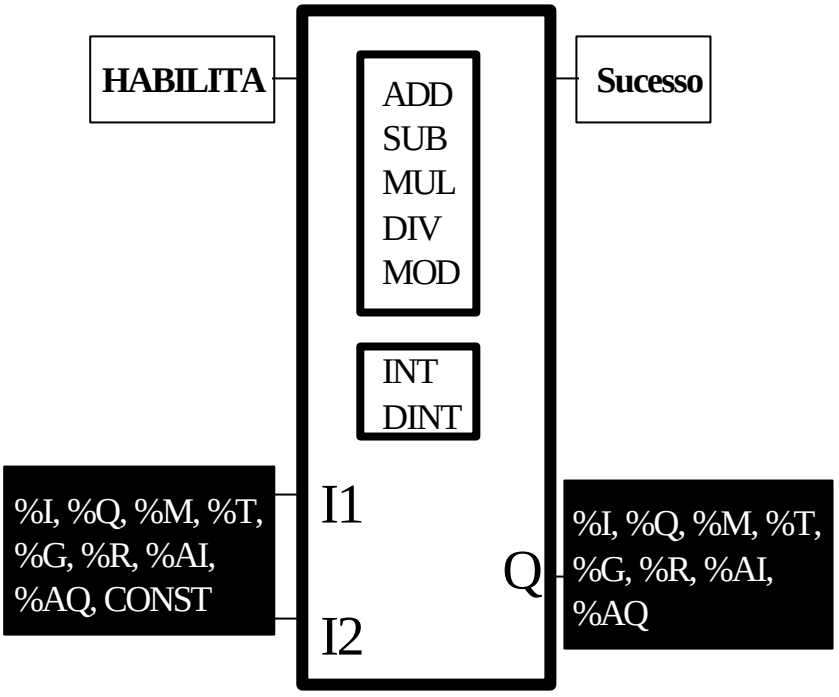


Figura 46 - Instruções de operações aritmética do GE FANUC

8.13.2 Instruções de comparação

Estas instruções permitem comparar valores numéricos e podem ser do tipo: maior, maior ou igual, menor, menor ou igual, igual, diferente e comparação entre limites. A **Erro! A origem da referência não foi encontrada.** ilustra os blocos funcionais de comparação do GE FANUC. Quando o bloco é habilitado o resultado lógico da comparação é apresentado na saída Q.

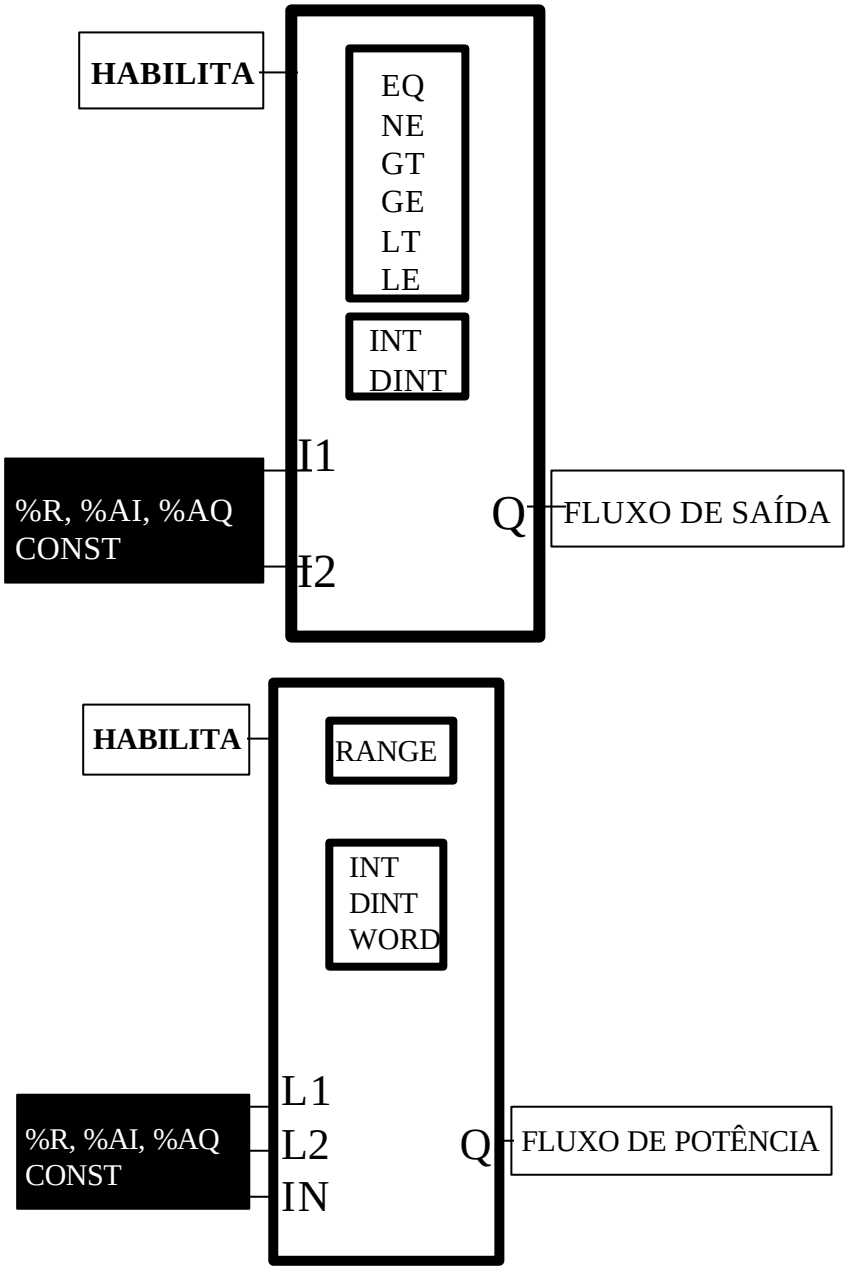


Figura 47 – Instruções de Comparação do CLP GE FANUC

O bloco compara sempre I1 em relação a I2. No bloco de comparação RANGE, L1 contém o valor inicial do intervalo. L2 contém o valor final do intervalo. Quando o valor contido em IN estiver entre L1 e L2 a saída Q será energizada.

8.13.3 Partida de motor com rampa de aceleração

Deseja-se acionar um motor através de um sistema CLP e inversor de freqüência. Para isto uma saída analógica do CLP é conectada na entrada de referência de velocidade no inversor de freqüência. O CLP possui um módulo de saída analógica de 0 a 10 V para uma variação interna do valor digital de 0 a 32000. Implemente um programa para criar uma rampa de aceleração considerando que a cada 2 s a saída analógica deverá ser incrementada de 1 V até atingir o valor máximo de 10 V.

Solução: (implementação para controladores GE-FANUC)

Página intencionalmente deixada em branco

Página intencionalmente deixada em branco

9 PRÁTICA COM O ISAGRAF

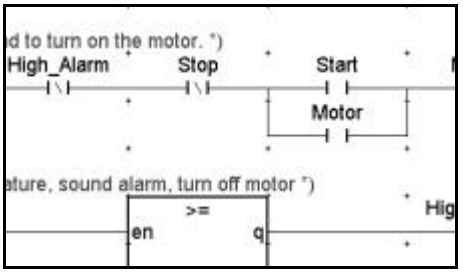
NESTA SEÇÃO SERÁ VISTO:

- A estrutura e os recursos do ISaGRAF
- Definição básica da terminologia ISaGRAF

O QUE É ISAGRAF?

ISaGRAF é um ambiente de programação que permite o desenvolvimento de uma aplicação de controle industrial baseada nas linguagens de programação normalizadas na IEC 61131-3.

AS LINGUAGENS DE PROGRAMAÇÃO IEC 61131-3



Diagramas Ladder (*Ladder Diagrams – LD*) é a representação clássica de lógica através de símbolos de contatos e bobinas de relés encontrada na maioria dos controladores programáveis. Diagramas ladder podem ser criados através do editor ***Quick Ladder***, de uma forma rápida e simplificada combinando elementos básicos para formar um programa.

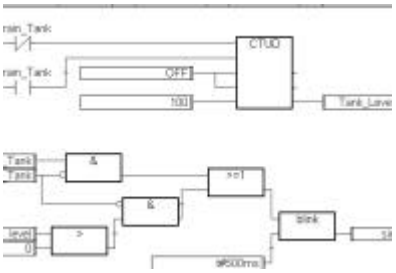
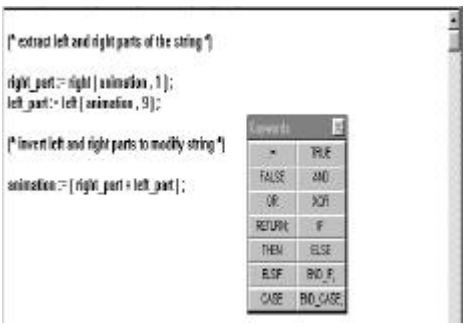


Diagrama de blocos funcionais (*Function Block Diagrams - FBD*) é a linguagem de programação encontrada em controladores de processo de alto desempenho e em sistemas DCS (sistemas de controle distribuído). Diagrama em blocos funcionais e diagramas ladder podem ser combinados em um único diagrama para obter funções de controle de máquinas e processos.

Diagrama de Funções Sequenciais (*Sequential Function Charts - SFC)* é uma linguagem gráfica poderosa que organiza o projeto e permite criar passo-a-passo operações sequenciais. Utiliza uma representação gráfica para os diferentes passos de um processo, ligado por condições booleanas chamadas de transição. Ações dentro dos passos são detalhadas através de outras linguagens (ST, IL, LD e FBD).



Texto Estruturado (*Structured Text - ST*) é uma linguagem de programação similar ao Pascal. Representa uma maneira eficiente de programar operações complexas. Texto estruturado é frequentemente utilizado para criar blocos funcionais definidos pelo usuário, para descrever passos e transições.

Lista de Instruções (*Instruction List - IL*) é uma linguagem composta por mnemônicos. É baseada na

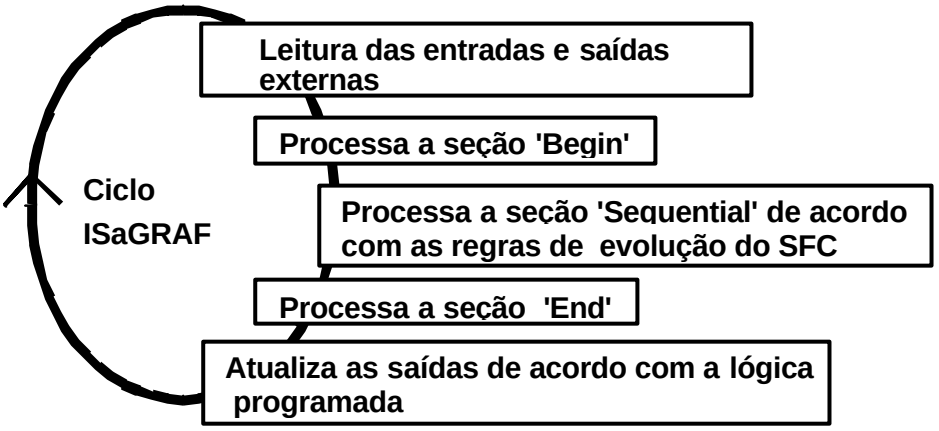
lista de instruções Siemens.

O QUE É PROJETO ISAGRAF ?

Um projeto ISaGRAF é uma coleção de programas e funções individuais que formam uma aplicação de controle completa. Cada programa controla uma seção particular da aplicação. Programas e funções são divididos em 4 seções dentro do projeto de acordo com sua posição no ciclo de varredura do ISaGRAF. Todas as 5 linguagens IEC 61131-3 podem ser utilizados dentro de uma mesma aplicação.

CICLO DE VARREDURA ISAGRAF:

Durante a execução do projeto ISaGRAF o programa de controle é executado de acordo com o seguinte ciclo:



SEÇÕS DE PROGRAMAÇÃO:

Como mencionado anteriormente, cada projeto ISaGRAF consiste de uma série de programas, sub-programas e funções que são divididas em 4 seções diferentes, de acordo com sua posição no ciclo ISaGRAF. Cada seção pode conter vários programas ou mesmo ser vazio. As seções são separadas por uma barra horizontal na janela de gerenciamento de programa ISaGRAF.

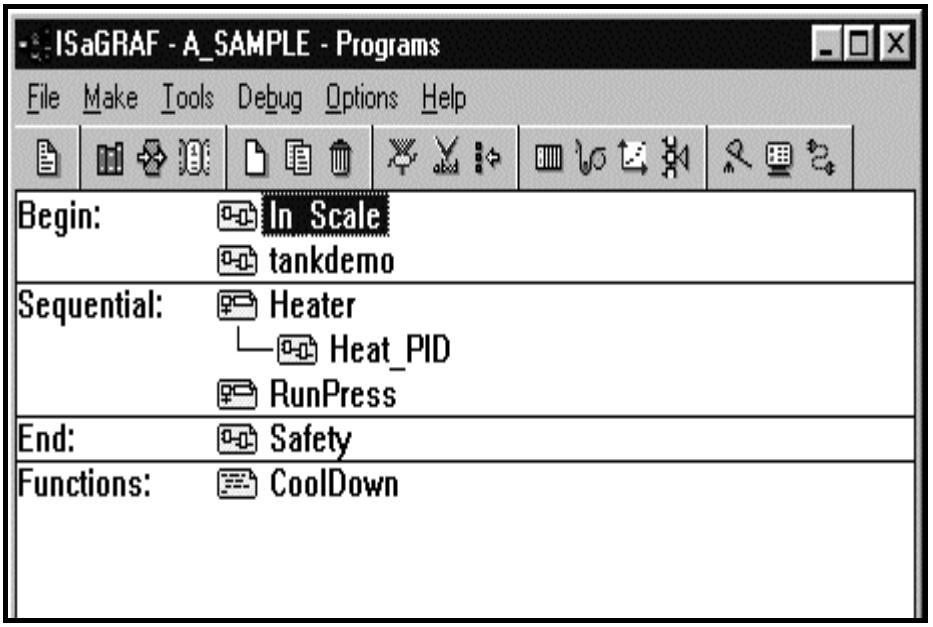
Beginning: Programas cíclicos que não dependem do tempo. Programas nesta seção são sistematicamente executados no começo do ciclo após a varredura das entradas externas. Esta seção não pode ser do tipo SFC.

Sequential: Esta seção destina-se a programas SFC. Suporta programação de processos paralelos e programas com relacionamento pai-filho.

End: Esta seção possui as mesmas características da seção **beginning,** porém é executada no final do ciclo antes da atualização das saídas externas.

Functions: São sub-programas que podem ser chamados por um programa em qualquer outra das três seções. Funções não podem ser SFC.

EXEMPLO DE UM PROGRAMA ISAGRAF:



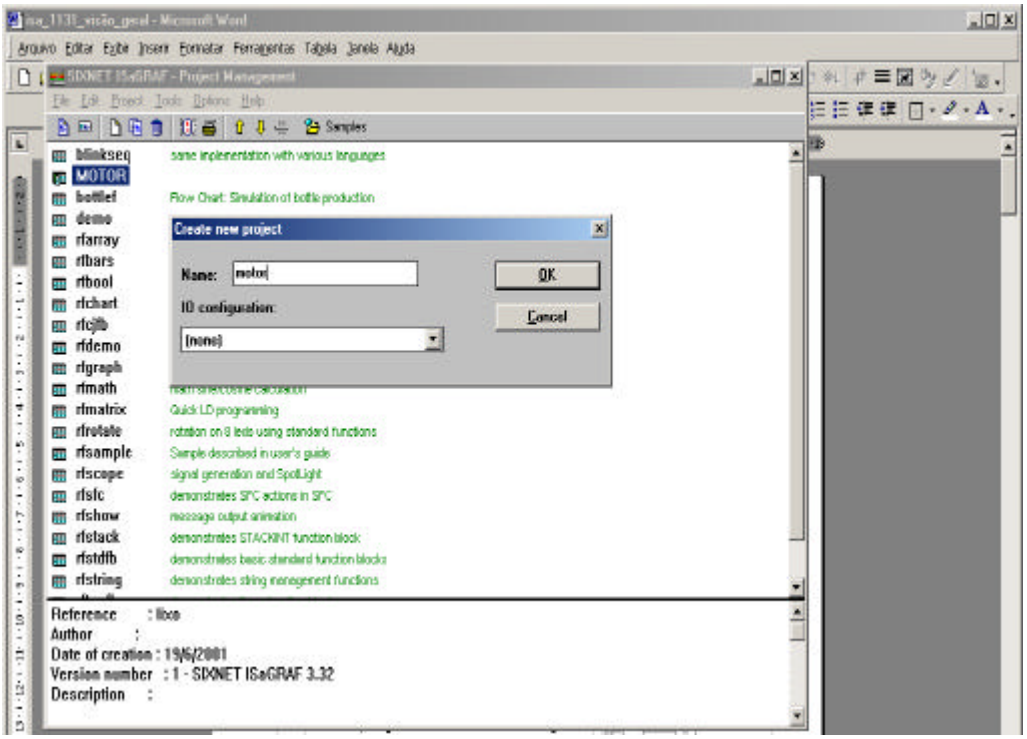
UTILIZANDO O PROGRAMA ISAGRAF

Passo 1:

Rodar o programa Isagraf através do menu Iniciar do Windows. Após a execução irá aparecer a janela do Isagraf chamada de Gerenciador de Projetos.

Passo 2:

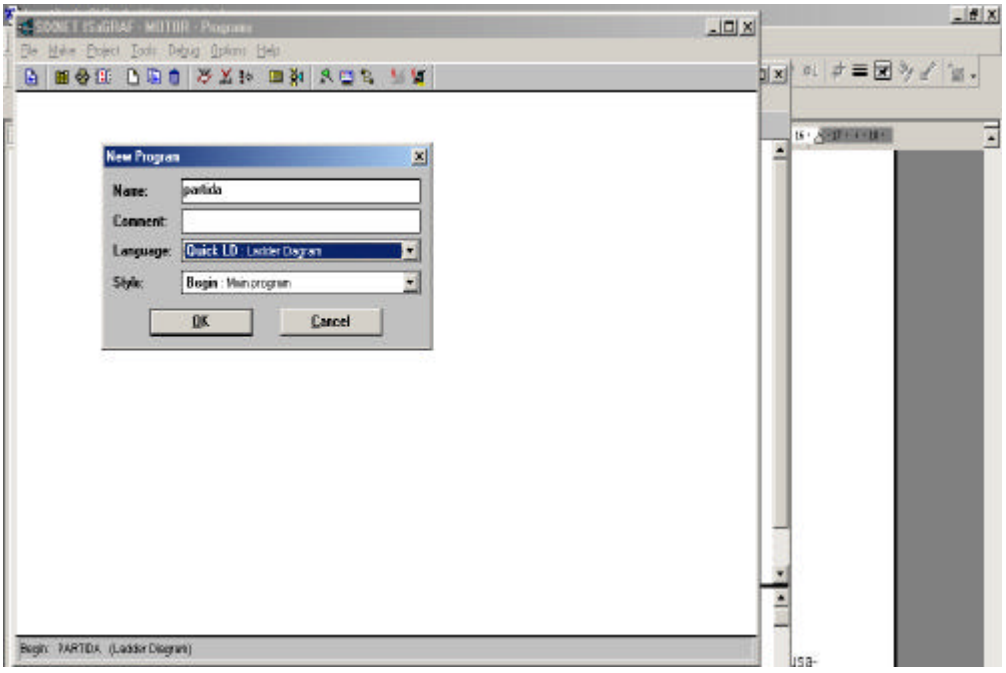
No menu **File**, criar um novo projeto através do sub-menu **New**. Observe a figura a seguir. Digite o nome do projeto a ser criado, por exemplo – motor, e clique em **OK**. Não é necessário preencher o campo **IO Configuration**.



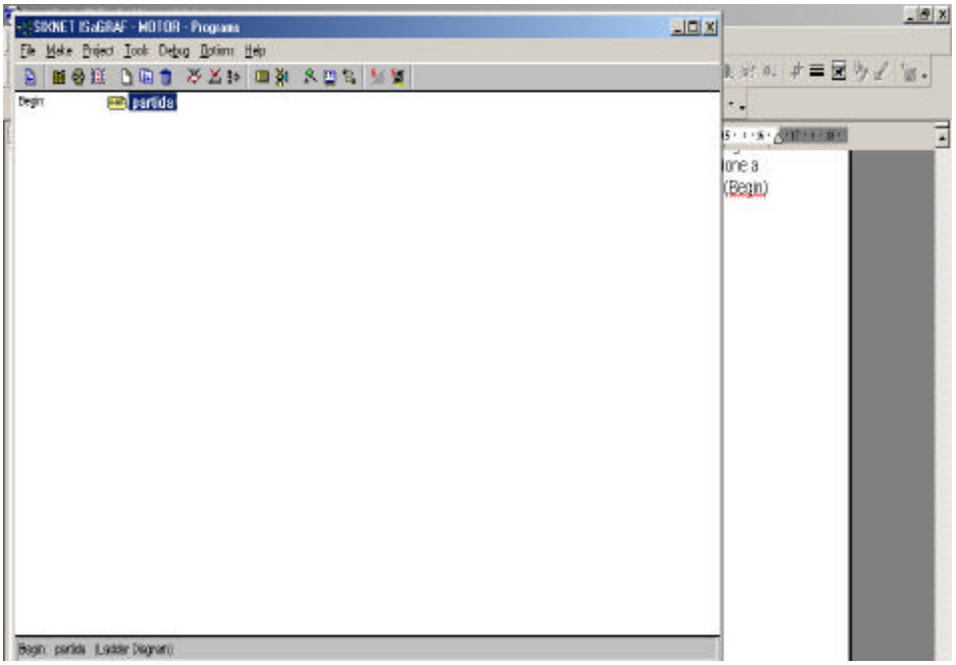
Passo 3:

Através do menu File na janela de programas, selecione o sub-menu New e crie um programa. Digite o nome do programa e selecione a linguagem que deseja utilizar, no caso selecione a linguagem Quick LD. Em seguida, selecione a seção onde deseja criar este programa (Begin)

Clique em Ok para fechar a janela e retornar a janela principal.



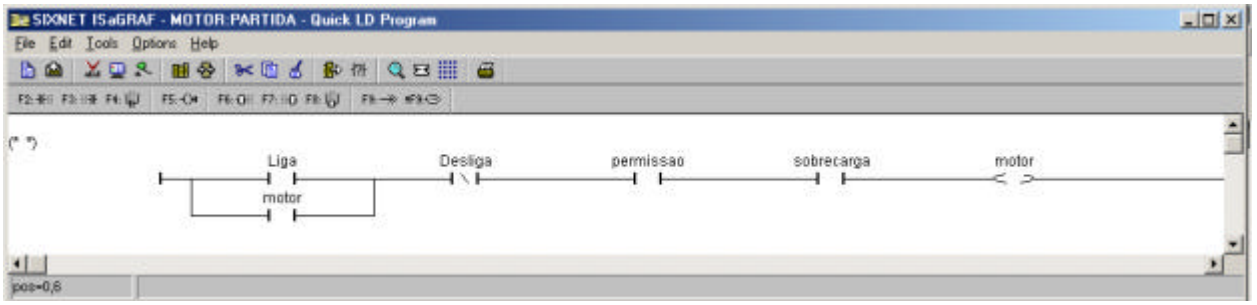
Passo 4:



Observe que foi criado o programa “partida” a ser desenvolvido em Ladder Diagram. De um duplo clique no ícone “partida” para editar o programa.

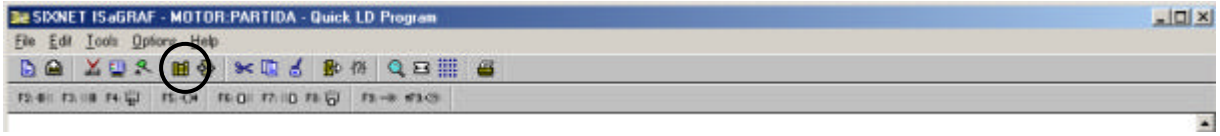
Passo 5: Edição do programa ladder

Utilizando as teclas F2, F3, ... ou através do mouse construa a lógica conforme desejado. Para inserir o nome da variável nas instruções, dê um duplo clique com o mouse sobre a instrução desejada. Ao aparecer a janela, digite o nome da variável.



Passo 6 Dicionário de variáveis

A criação do dicionário é realizada através do ícone sinalizado abaixo.



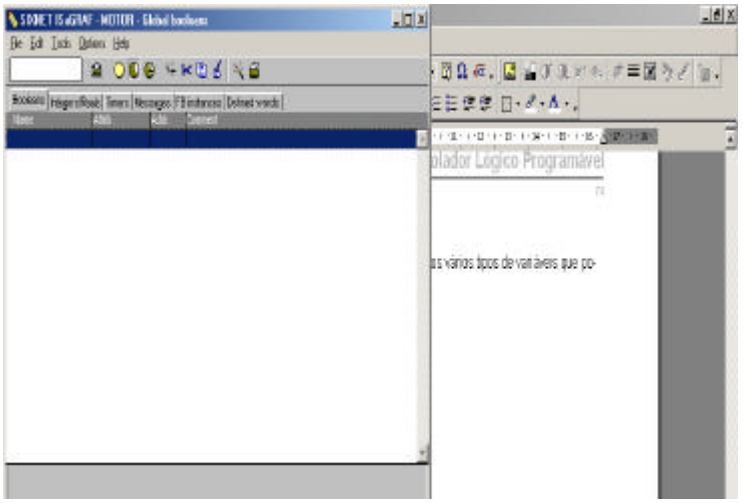
O dicionário do ISaGRAF relaciona as variáveis internas e externas bem como as definições usadas no programa de um projeto **(defines)**. Durante o desenvolvimento da aplicação, o usuário utiliza o dicionário para acessar as variáveis necessárias ao projeto.

Passo 7: Criação das variáveis booleanas

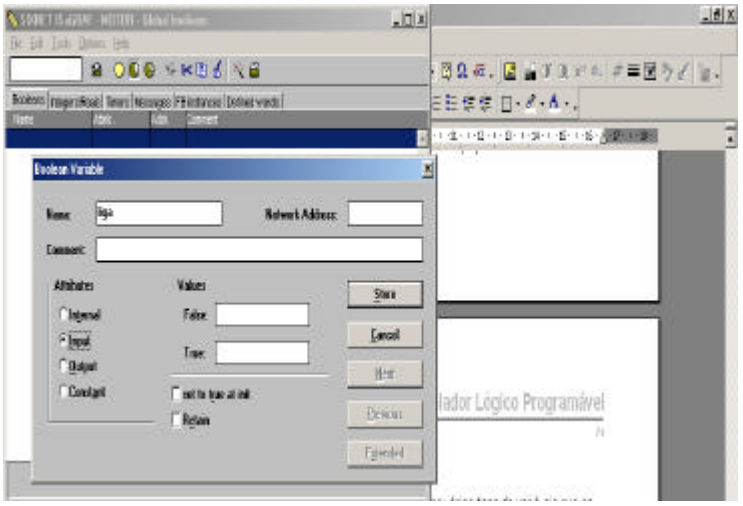
Ao clicar no ícone a seguinte janela irá aparecer. Observe os vários tipos de variáveis que podem ser criadas. As variáveis podem ser do tipo:

- Booleanas
- Reais/inteiras
- Temporizadas
- Mensagens
- Instância de blocos
- Palavras definidas

Selecione **variáveis booleanas** (provavelmente já estará selecionada por *default*)



Posicione o mouse na janela central e clique o botão da direita. A seguinte janela irá aparecer. Digite o nome da variável que deseja criar **observando se o atributo da mesma, interna, entrada, saída ou constante.**



Repita este procedimento para todas as variáveis.

Variáveis internas são nomes lógicos de elementos do programa que são necessários para computação ou análise interna. Por exemplo, pode ser necessário criar um variável interna para armazenar o valor de engenharia de uma grandeza analógica, tal como temperatura de um forno.

Definições (**define)** permite ao usuário a utilizar um nome descritivo para representar um valor constante. Este recurso facilita a manutenção do sistema. Como o nome descritivo pode ser utilizado em todo o projeto, quando se deseja alterar o seu valor basta alterar no dicionário e os efeitos se propagarão em todo o projeto.

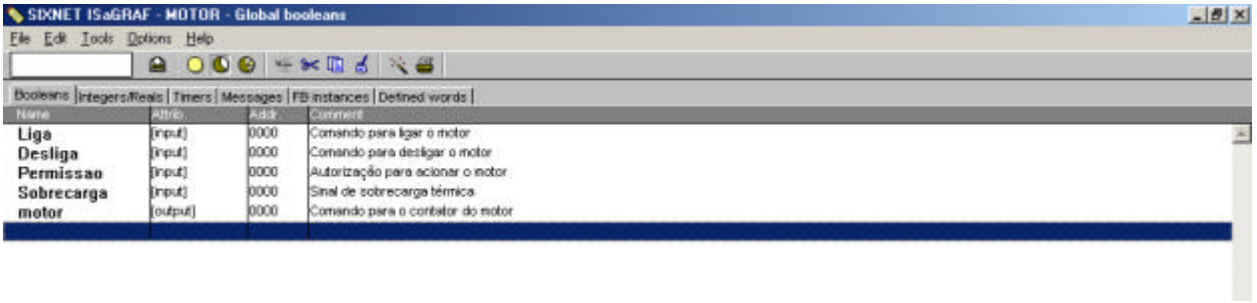
As variáveis e definições podem ser criadas no dicionário a medida que o projeto vai sendo concebido e podem ser do tipo:

- **Local** quando pertencer ao escopo do programa
- **Global** quando utilizado em qualquer programa dentro de um projeto
- **Common** quando utilizado em qualquer projeto ISaGRAF

OBSERVAÇÃO IMPORTANTE:

Digitar o nome da variável, o comentário (opcional) e selecione o atributo adequadamente. Marque **INPUT** para as entradas digitais e **OUTPUT** para as saídas digitais. O atributo **INTERNAL** é utilizado para criar variáveis internas. Os outros campos não são necessários preencher.

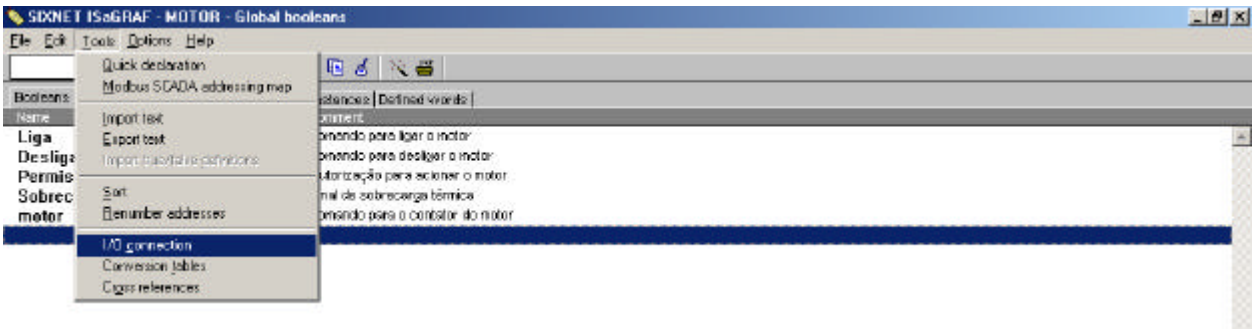
A tela final após a criação das variáveis será:



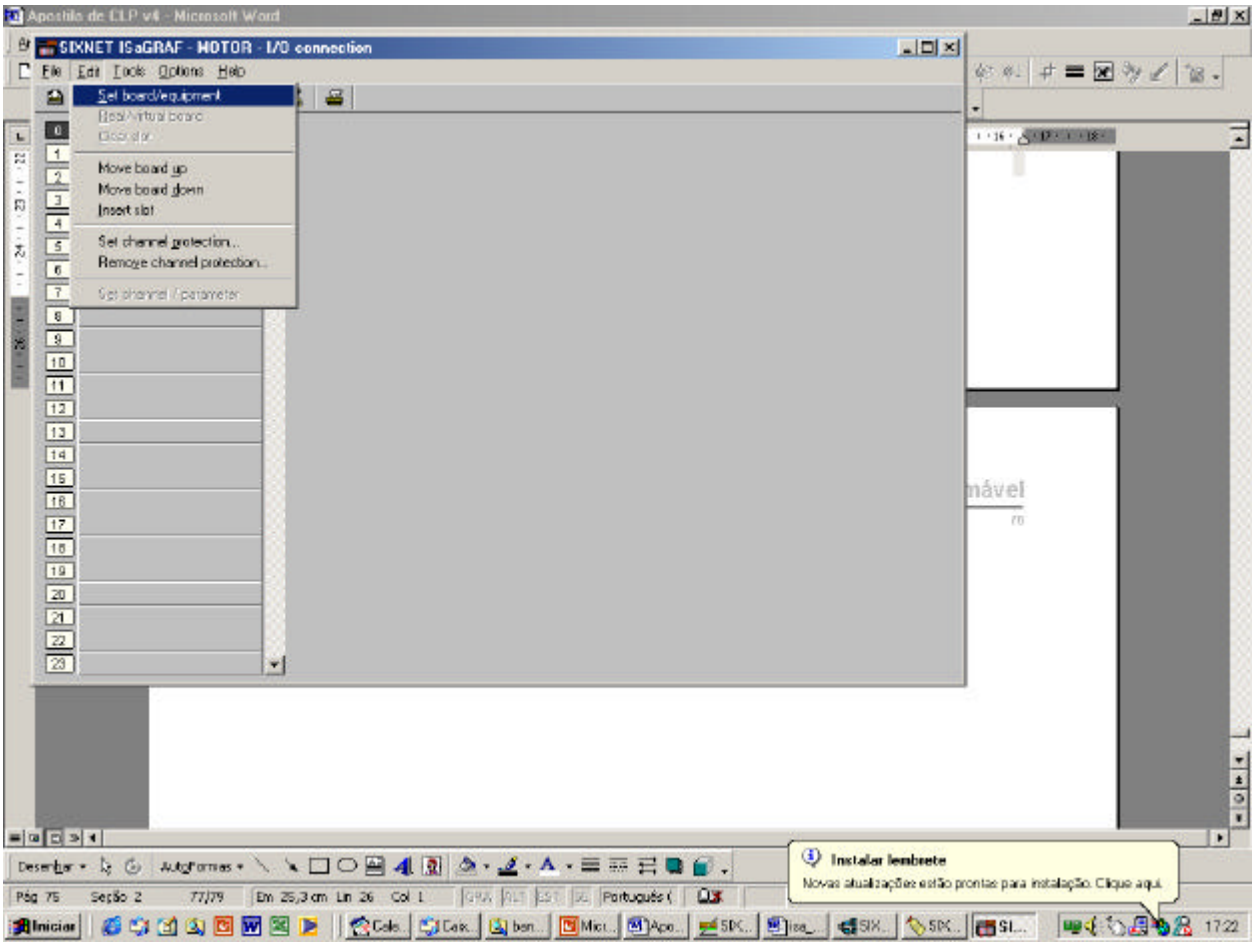
Passo 8: Conexão das Entradas e Saídas

Após a criação das variáveis é necessário realizar a conexão das mesmas. Este procedimento é extremamente necessário e permitirá a simulação do programa.

Na janela do dicionário, clique no menu **Tools** e selecione o sub-meu **I/O connection**.

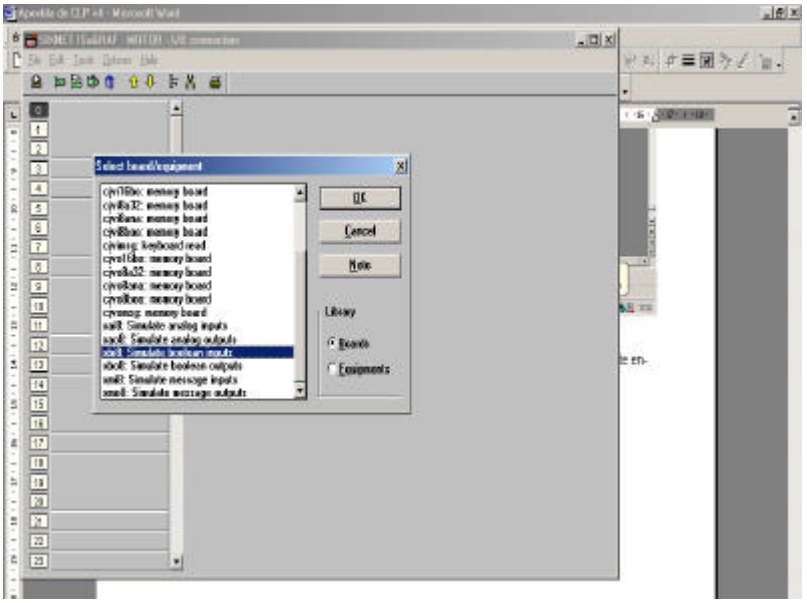


Na janela de IO connection, selecione o sub-menu **Edit** e em seguida selecione o sub-menu **Set board/equipment** .

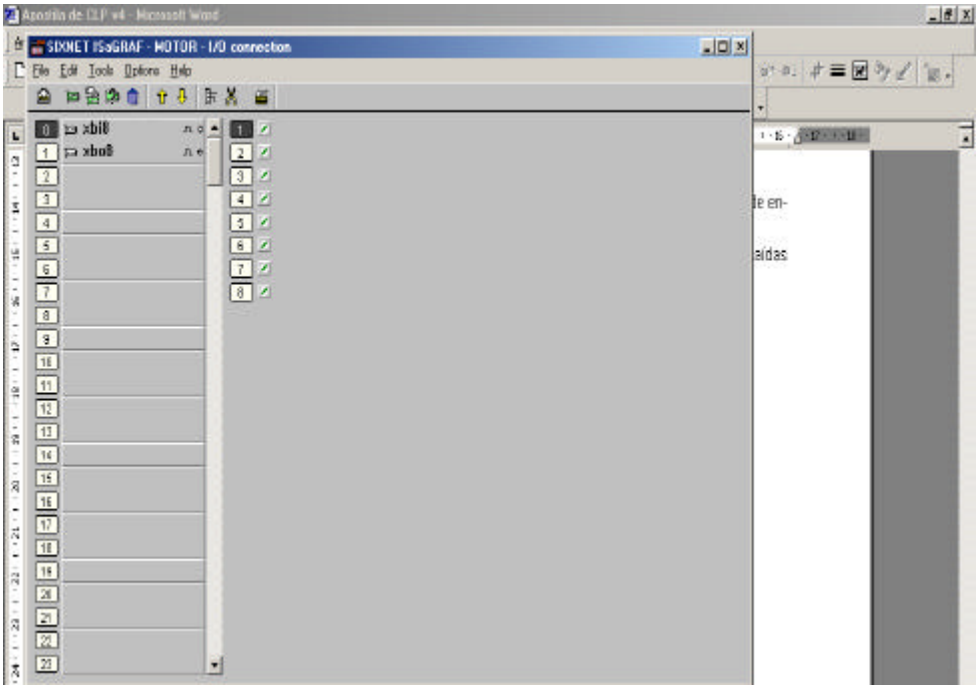


Após clicar o sub-menu Set board/equipment, selecione na janela seguinte os módulos de entradas e saídas externas que desejar.

Para o exercício em questão, criar um módulo de entradas booleanas e um módulo de saídas booleanas.

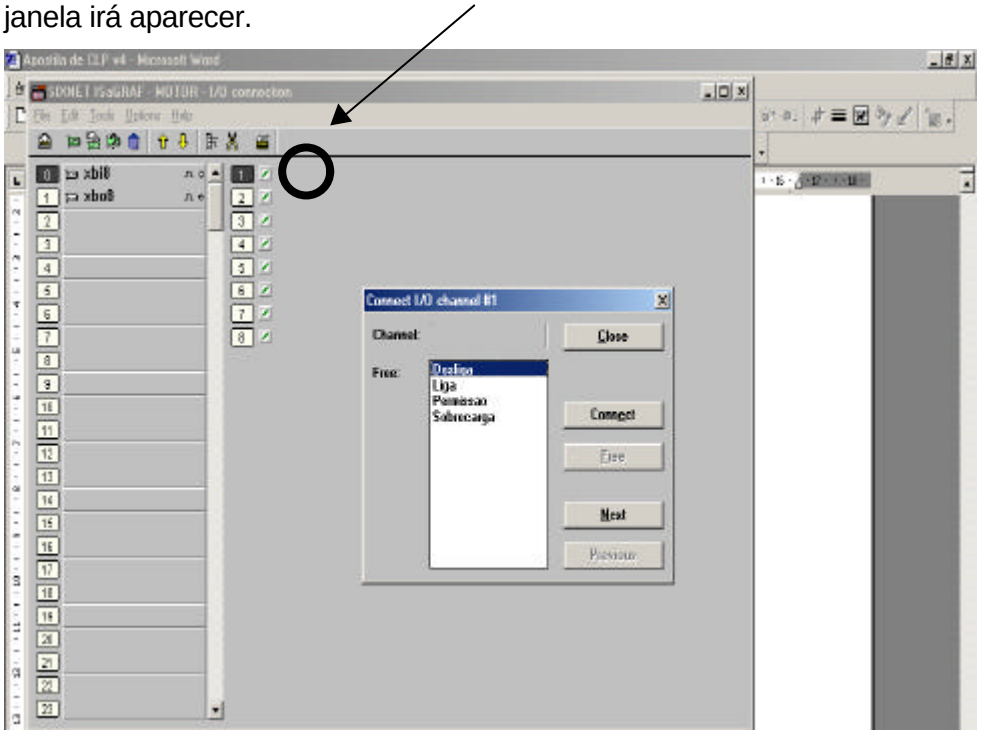


Ao final deste passo a sua janela deve estar da seguinte forma:

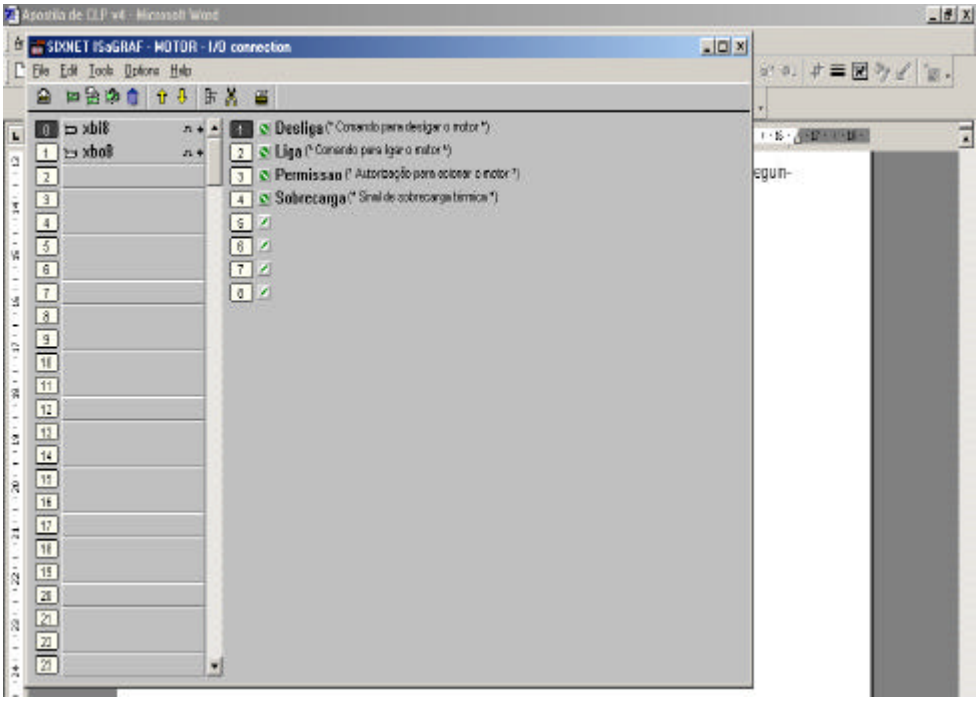


Passo 9: Simulação da conexão física das entradas

Dê um duplo clique em cima do slot 1 na janela da direita conforme ilustrado abaixo. A seguinte janela irá aparecer.



Nesta janela clique em **Connect** tantas vezes quanto for necessário para simular a conexão física das entradas. Ao final sua tela deverá ser:

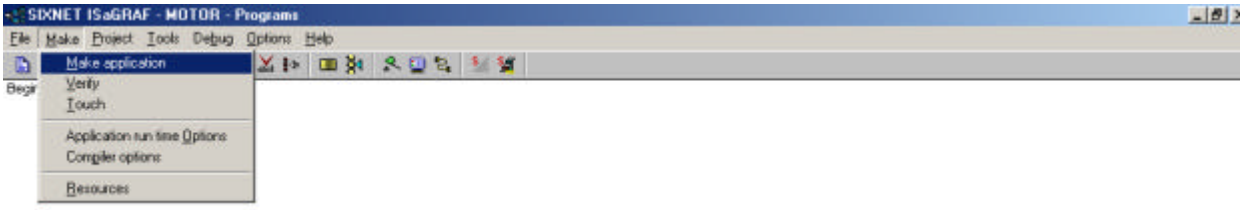


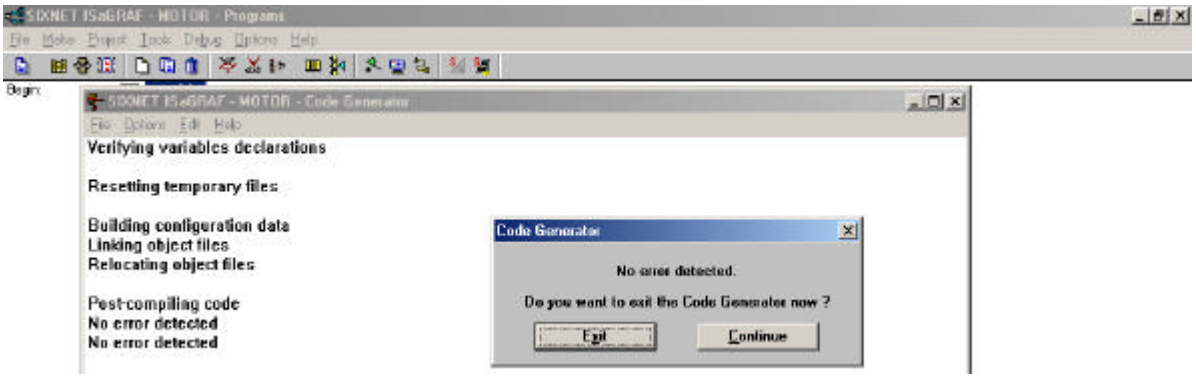
Passo 10: Conexão física das saídas

Repita o procedimento anterior considerando o módulo de saída.

Passo 11: Compilação do programa

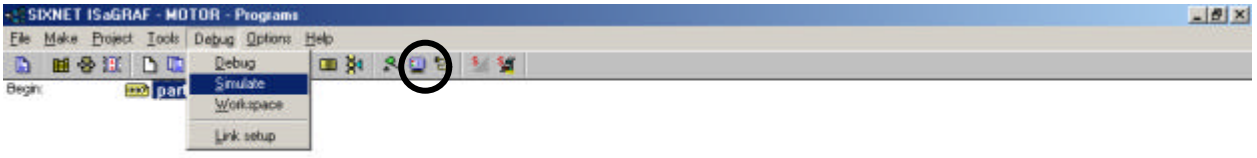
Feche todas as janelas do dicionário e salve todas as alterações realizadas. Na janela principal, gerenciador de programas, selecione o menu **Make** e em seguida o sub-menu **Make application**. Esta operação realiza a compilação do programa gerado e se o programa estiver livre de erros uma janela de mensagem será apresentada.





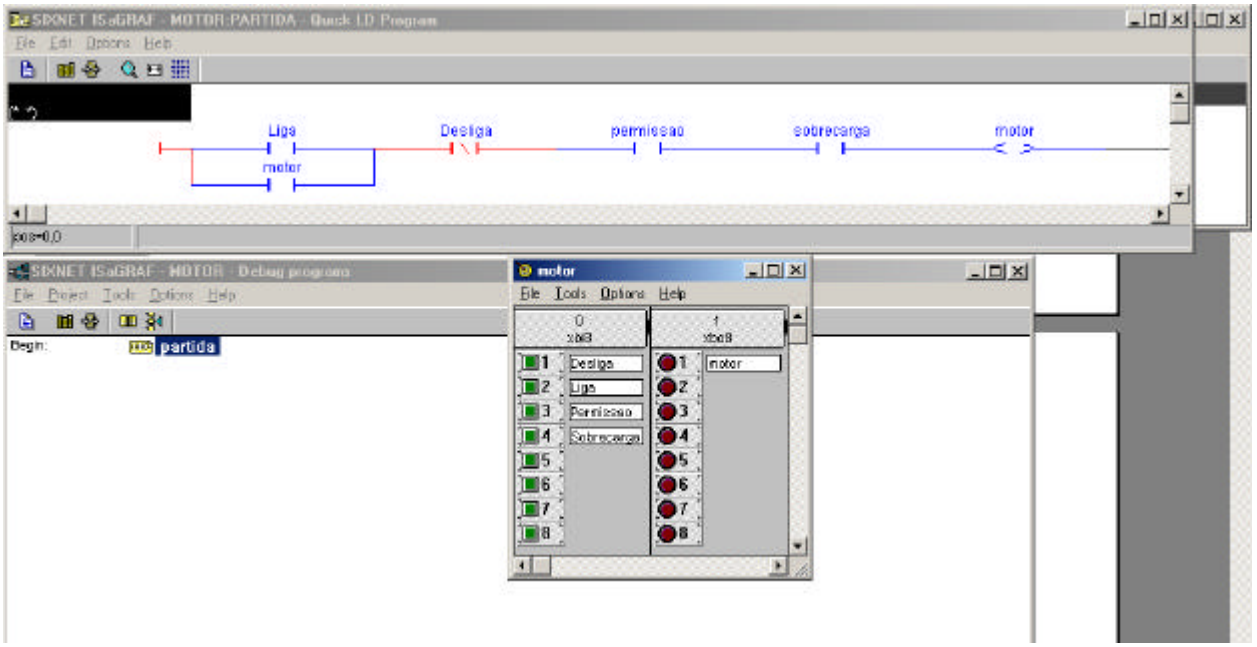
Passo 12: Rodando a simulação do programa

Clique no menu **Debug** e em seguida no sub-menu **Simulate** ou através do ícone destacado abaixo.



Passo 13: Painel de simulação

Organize a tela de forma que você possa ver as telas de interesse. O painel de simulação irá aparecer automaticamente juntamente com uma tela indicando que o programa está em execução. Na tela ***Debug programs*** dê um duplo clique no ícone partida para obter a tela de visualização on-line do programa ladder.



OBS:

1) Com o botão esquerdo do mouse acione as entradas de forma retentiva clicando no sinalizador ao lado do nome da variável da entrada
2) Com o botão direito do mouse o efeito nas entradas é do tipo instantâneo (não retentivo)

PROPOSTA DE EXERCÍCIOS PRÁTICOS NO ISAGRAF

1) Implementar o exercício de partida de motor utilizando Lista de instruções.

2) Implementar o exercício de partida de motor utilizando Texto Estruturado

3) Implementar o exercício das lâmpadas seqüências utilizando Lista de instruções

4) Implementar o exercício das lâmpadas seqüenciais utilizando texto estruturado

5) Implementar um programa que converta para unidade de engenharia considerando a seguinte relação: 0 a 32767 para 0 a 300 °C.